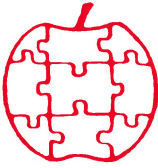


Apple

\$1.80



Assembly

Line

Volume 7 -- Issue 2

November, 1986

In This Issue...

Ultra-Fast Integer Square Roots.	2
New ProDOS Bug and Fix	13
Timing Apple Programs with Another Apple	15
The //gs Reference Manuals	19
PAUSE Directive for S-C Macro Assembler 2.0.	20
//gs Battery RAM and Clock/Calendar.	24

Some Price Reductions...

The other day I noticed that I could buy ten 3M diskettes in a nice hard-plastic library case for less than \$11 at the local Safeway store. Wow! Times have changed! Our price keeps going down too, though. Now you can buy disks from us for 60 cents apiece. A shrink-wrapped pack of 25 is only \$15, including tyvek sleeves.

We told you a few months ago about the Minuteman UPS from Para Systems. I still love mine, and I think you would also enjoy one as much as I do. We are lowering our price this month from \$350 to \$320, plus shipping charges. The Minuteman handles up to 250 watts. I run my Sider, printer, monitor, and //e with a full deck of cards including 1Meg RAMWORKS; there is probably ample power left for a few more items. My power is now filtered, surge protected, brownout protected, and blackout protected.

Ultra-Fast Integer Square Roots.....Charles H. Putney
and Bob Sander-Cederlof

Well, Bob wanted a faster integer square root program, so here it is! This method uses table lookup with as little as two pages of tables. The fastest version uses 2.75 pages of tables (704 bytes), and averages only 37 microseconds per root when taking all 65536 possible. The version which uses only 512 bytes of tables is a little slower, but still a lot faster than the IBM PC program Bob mentioned a few months ago.

Here's how my method works. First the input argument is shifted left two bits at a time until it is in the range from \$4000 to \$FFFF. I keep track of how many double-bit shifts this takes, from 0 to 7 times. Then I use the high byte of this value, which will be a number from \$40 to \$FF, to find the root in a table of 192 roots. Then I shift the root right from 0 to 7 times, depending on the number of shift steps used before. The result is either the correct integer square root of the original number, or one less than the correct root. I can make the final correction by testing the original argument against a table of squares. I use the root taken from the first table as index into the second table. The value in the second table for root=N will be $(N+1)*(N+1)$. If the original argument is less than N-plus-1-squared, then N is the correct root; otherwise, N+1 is the correct root.

The program is shown below, with the tables. I used an Apple-soft program to actually generate the data for the tables, in a form which can be EXECed directly into the S-C Macro Assembler:

```
10  D$ = CHR$(4)
20  PRINT  CHR$(4);"OPEN SQUARE ROOT TABLE"
30  PRINT  CHR$(4);"WRITE SQUARE ROOT TABLE"
40  H$ = "0123456789ABCDEF"
100 REM TABLE OF ROOTS
110 PRINT "5000 TABLE1"
120 FOR I = 0 TO 23
130 PRINT 5001+I "    >HS ";
140 FOR J = 64 TO 71
150 N = (I*8+J) * 256
160 R = INT (SQR(N))
170 GOSUB 1000
180 NEXT : PRINT : NEXT

200 REM TABLE OF SQUARES (LOW BYTES)
210 PRINT "6000 TABLE2"
220 FOR I = 0 TO 31
230 PRINT 6001+I "    >HS ";
240 FOR J = 1 TO 8
250 N = I*8+J : N2 = N*N
260 R = N2 - 256*INT(N2/256)
270 GOSUB 1000
280 NEXT : PRINT : NEXT

300 REM TABLE OF SQUARES (HIGH BYTES)
310 PRINT "7000 TABLE3"
320 FOR I = 0 TO 31
```

S-C Macro Assembler Version 2.0.... DOS \$100, ProDOS \$100, both for \$120
 Version 2.0 DOS Upgrade Kit for 1.0/1.1/1.2 owners.....\$20
 ProDOS Upgrade Kit for Version 2.0 DOS owners... ..\$30
 Source Code of S-C Macro 2.0 (DOS only).additional \$100
 Full Screen Editor for S-C Macro (with complete source code).\$49
 S-C Cross Reference Utility.... without source code \$20, with source \$50
 RAK-Ware DISASM. without source code \$30, with source \$50
 S-C Word Processor (with complete source code).\$50
 DP18 Source and Object.... ..\$50
 Double Precision Floating Point for Applesoft (with source code).....\$50
 ES-CAPE (Extended S-C Applesoft Program Editor).
 Including Version 2.0 With Source Code... ..\$50
 ES-CAPE Version 2.0 and Source Code Update (for Registered Owners)...\$30
 Bag of Tricks 2 (Quality Software).(\$49.95) \$45 *
 MacASM -- Macro Assembler for Macintosh (Mainstay). . .(\$150.00) \$100 *
 S-C Documentor (complete commented source code of Applesoft ROMs)....\$50
 Cross Assemblers for owners of S-C Macro Assembler... \$32.50 to \$50 each
 (Available: 6800/1/2, 6301, 6805, 6809, 68000, Z-80, Z-8, 8048,
 8051, 8085, 1802/4/5, PDP-11, GI1650/70, others)

AAL Quarterly Disks.... ..each \$15, or any four for \$45
 Each disk contains the source code from three issues of AAL:
 Jan-Mar, Apr-Jun, Jul-Sep, and Oct-Dec.
 (All source code is formatted for S-C Macro Assembler. Other assemblers
 require some effort to convert file type and edit directives.)

Diskettes (with hub rings). package of 25 for \$15 *
 Vinyl disk pages, 6"x8.5", hold two disks each... ..10 for \$6 *
 Diskette Mailing Protectors (hold 1 or 2 disks).....40 cents each
 (Cardboard folders designed to fit 6"x9" Envelopes.) or \$25 per 100 *
 Envelopes for Diskette Mailers... .. 6 cents each

65802 Microprocessor (Western Design Center).... (\$95) \$50 *
 quikLoader EPROM System (SCRG). (\$179) \$170 *
 PROMGRAMMER (SCRG). (\$149.50) \$140 *
 Switch-a-Slot (SCRG). (\$179.50) \$170 *
 Minuteman 250 Uninterruptible Power Supply... .. (\$359) \$320 +

"65816/65802 Assembly Language Programming", Fischer..... (\$19.95) \$18 *
 "Programming the 65816", Eyes.. (\$22.95) \$21 *
 "Apple //e Reference Manual", Apple Computer... .. (\$24.95) \$23 *
 "Apple //c Reference Manual", Apple Computer... .. (\$24.95) \$23 *
 "ProDOS Technical Reference Manual", Apple Computer. (\$29.95) \$27 *
 "Apple ProDOS: Advanced Features for Programmers", Little.. (\$17.95) \$17 *
 "Inside the Apple //c", Little. (\$19.95) \$18 *
 "Inside the Apple //e", Little. (\$19.95) \$18 *
 "Apple II+/IIe Troubleshooting & Repair Guide", Brenner. .. (\$19.95) \$18 *
 "Apple II Circuit Description", Gayler.... .. (\$22.95) \$21 *
 "Understanding the Apple II", Sather. (\$22.95) \$21 *
 "Understanding the Apple //e", Sather... .. (\$24.95) \$23 *
 "Enhancing Your Apple II, vol. 1", Lancaster... .. (\$15.95) \$15 *
 "Enhancing Your Apple II, vol. 2", Lancaster. (\$17.95) \$17 *
 "Assembly Cookbook for the Apple II/IIe", Lancaster..... (\$21.95) \$20 *
 "Beneath Apple DOS", Worth & Lechner. (\$19.95) \$18 *
 "Beneath Apple ProDOS", Worth & Lechner.. (\$19.95) \$18 *
 "Real Time Programming -- Neglected Topics", Foster.... (\$9.95) \$9 *
 "Microcomputer Graphics", Myers... .. (\$14.95) \$14 *
 "Assem. Language for Applesoft Programmers", Finley & Myers. (\$18.95) \$18 *
 "Assembly Lines -- the Book", Wagner. (\$19.95) \$18 *
 "AppleVisions", Bishop & Grossberger... .. (\$39.95) \$36 *

* On these items add \$2.00 for the first item and
 \$.75 for each additional item for US shipping.

+ Inquire for shipping cost.

Foreign customers inquire for postage needed.

Texas residents please add 6 1/8 % sales tax to all orders.

*** S-C SOFTWARE, P. O. BOX 280300, Dallas, TX 75228 ***
 *** (214) 324-2050 ***
 *** Master Card, VISA, Discover and American Express ***

```

330 PRINT 7001+I "    >HS ";
340 FOR J = 1 TO 8
350 N = I*8+J : N2 = N*N
360 R = INT(N2/256) : IF R>255 THEN R=0
370 GOSUB 1000
380 NEXT : PRINT : NEXT

900 PRINT D$"CLOSE"
910 END

1000 REM PRINT IN HEX
1010 PRINT MID$(H$, INT (R / 16) + 1,1);
1020 PRINT MID$(H$,R - 16 * INT (R / 16) + 1,1)". ";
1030 RETURN

```

The way I have written the SQRT subroutine, the high byte of the argument is expected in the X-register and the low byte in the A-register. The square root is returned in the Y-register, with A and X destroyed. This combination seemed to me to give the best speed. Lines 2290 and 2300 divide the arguments into two ranges: \$4000-FFFF, and below \$4000. The higher range comprises 75% of the possible arguments, a total of 49152.

The top 256 possible arguments, from \$FF00 to \$FFFF, must be handled as a special case. The logic which compares roots against values in TABLE2 and TABLE3 is confused by the fact that the entry for \$FF is \$0000. (It really is \$10000, but the leading 1 is not in either table.) Lines 2320-2330 strip out these arguments, and lines 2860-2870 return the correct root (\$FF). A total of only 11 cycles (not counting JSR SQRT or RTS) for these 256 arguments.

If the range is from \$4000 to \$FEFF, as it is in 48896 cases, lines 2340-2410 return the correct root. The high-byte of the argument is already in the X-register, so line 2340 loads the Y-register with the trial root from the table. No shifting must be done, so lines 2350-2390 proceed to compare with the square of the root+1 in TABLE2 and TABLE3. If the entry there is larger than the original argument, the root is correct; if not, line 2400 adds one, making it correct. The longest path from beginning to end for these arguments is only 28 cycles. If the test at line 2360 branches, it is only 19 cycles. Wow! And this takes care of three-fourths of all cases!

Arguments below \$4000 are handled by lines 2430 and following. Lines 2440-2450 test for arguments from \$0000 to \$00FF. These will be handled by lines 2740-2840. If the argument is exactly \$0000, the root is \$00, and this is detected at lines 2740-2750. All other roots below \$0100 need to be shifted at least 4 two-bit steps. By merely starting to work on the low-byte, and with a shift-count of 4, we accomplish the first four steps without taking any time at all. The loop in lines 2790-2840 normalizes the byte and continues to count shift steps. Then we join the processing of values between \$0100 and \$3FFF.

The range of arguments from \$0100 to \$3FFF are handled beginning at line 2470. The loop in lines 2510-2570 normalizes the argument by shifting left in two-bit steps until the value

is \$4000 or more, counting the number of steps it takes. It will be 1, 2, or 3 steps.

We come to line 2590 with a shift count in the Y-register. The count will be 1, 2, or 3 the original argument was \$0100 or more; it will be from 4 to 7 if the original argument was below \$0100. We also come to line 2590 with the high-byte of the normalized argument in the A-register. Lines 2590-2600 used this byte to get a trial root from TABLE1. The loop in lines 2610-2630 shifts the trial root right the same number of bits as we took in two-bit steps to normalize the argument earlier. Finally, lines 2640-2720 check the trial root against the value in TABLE2 and TABLE3, and correct the root if necessary.

I had it all counted out at one time, and the arguments below \$4000 (with the exception of \$0000) take on the order of 80 cycles. It gets very involved to try to count these paths, so I wrote a timing program instead. Lines 2070-2260 call the SQRT subroutine for each argument from \$0000 to \$FFFF, and do it ten times. This takes about 41 seconds to execute. For fun, I inserted line 2170 to toggle the speaker after taking each square root; the sound is interesting, and also reveals the fact that lower roots take longer than higher roots. Lines 2130 and 2250 turn on and off the AN0 signal in the game port, for the timing setup I describe in another article in this newsletter. Line 2220 makes a visible mark on the screen so I will not get too impatient while the program is running.

I ran the timing loop as shown first, and as I said it took about 41 seconds. My timing setup using another Apple to count cycles gave a result of 41,821,940 cycles. Then I changed line 2280 from "SQRT" to "SQRT RTS", so I could time the overhead of the the timing program itself. My other Apple said this took 17,056,520 cycles. The difference is the time of SQRT itself, and this is 24,765,420. Remember that I took 65536 square roots ten times: therefore I divide by 655360 to get an average cycle count of only 37.8 cycles. In English, that is about 37 microseconds. Wow! Tell that to your IBM friend!

The program slows down if the tables are not properly placed in memory. Indexed instructions take an extra clock cycle if the indexing crosses a page boundary. Therefore, I adjusted the start of the tables so that they fit in a page. Notice that TABLE1 really starts 64 bytes into the page. This is so because the index we use to access TABLE1 runs from \$40 to \$FF.. The label ROOT is equated to TABLE1-64, at line 4010.

All this timing is irrelevant if the program produces incorrect results. Therefore an exhaustive test is necessary. I wrote a test program in Applesoft, but it was very slow. Therefore I converted it to assembly language, with the result in lines 1340-2050. The test program has some interesting wrinkles in it. It checks all the square roots from SQRT without actually having any code to multiply, divide, or take a square root.

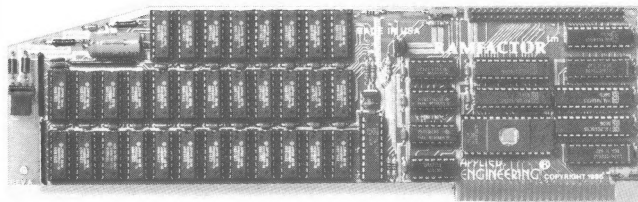
The test program runs through the possible arguments in sequential order, from \$0000 to \$FFFF. If the answer returned by SQRT is correct, it will pass the following tests:

RamFactor™

All the Performance, Speed, and Software Compatibility of RamWorks™ in a Slot 1 through 7 Card.

That's right! Now Applied Engineering offers you a choice. While RamWorks is the clear winner for the auxiliary slot in a IIe, RamFactor is the standard for slots 1 through 7. Now anyone with an Apple II+, Franklin, or Apple IIe preferring to use slots 1 through 7 can now enjoy the speed and performance that until now was only available with RamWorks.

With RamFactor, you'll be able to instantly add another 256K, 512K, or a full 1 meg on the main board and up to 16 meg with additional piggyback card. And since virtually all software is automatically compatible with RamFactor, you'll immediately be able to load programs into RamFactor for instantaneous access to information. You'll also be able to store more data for larger word processing documents, bigger data bases, and expanded spreadsheets.



Very Compatible

All the leading software is already compatible with RamFactor. Programs like AppleWorks, Pinpoint, BPI, Managing Your Money, Dollars and Sense, SuperCalc 3A, PFS, Mouse Write, MouseDesk, MouseCalc, Sensible Speller, Applewriter IIe, Business Works, ReportWorks, Catalyst 3.0 and more. And RamFactor is fully ProDos, DOS 3.3, Pascal 1.3 and CP/M compatible. In fact, no other memory card (RamWorks excepted) is more compatible with commercial software.

AppleWorks Power

There are other slot 1-7 cards that give AppleWorks a larger desktop, but that's the end of their story. But RamFactor is the only slot 1-7 card that increases AppleWorks internal memory limits, increasing the maximum number of records in the database and lines permitted in the word processor, and RamFactor is the only standard slot card that will automatically load AppleWorks into RAM dramatically increasing speed and eliminating the time required to access the program disk; it will even display the time and date on the AppleWorks screen with any ProDos clock. RamFactor will automatically segment large files so they can be saved on 5¼", 3½", and hard disks. All this performance is available to anyone with an Apple IIe or II+ with an 80

column card. RamFactor, no other standard slot card comes close to enhancing AppleWorks so much.

True 65C816 16 Bit Power

RamFactor has a built-in 65C816 CPU port for direct connection to our IIe 65C816 card for linearly addressing up to 16 meg for the most powerful 16 bit applications (II+ 65C816 card under development)

Powerful Program Switcher

With RamFactor, you can organize memory into multiple work areas and switch between them. Each work area can contain different programs and even different operating systems. Now you can switch from one program to another or even switch from AppleWorks to DOS 3.3 to CP/M to Pascal to ProDos in under a second. And with our Battery back-up option, you can have permanent storage for up to 20 years.

Quality and Support of the Industry Leader

RamFactor is from Applied Engineering, the largest, most well supported manufacturer of Apple peripherals and the inventor of large RAM cards for the Apple. With our 5 year no hassle warranty and outstanding technical support, you're assured of the most trouble free product you can buy.

Features:

- Up to 16 meg total memory, 256K to 1 meg on main board. Up to 16 meg with additional memory on piggyback card
- Fully Apple II Memory Expansion compatible
- Compatible with Apple IIe, II+ and Franklin
- Battery back-up option allows you to turn on your Apple and run your favorite programs in less than 1 second!
- Automatically recognized by ProDos, DOS 3.3, Pascal and CP/M
- Built-in RamDrive™ software (a true RAM disk not disk caching)
- Systems are directly bootable from RamFactor if desired
- Built-in linear addressing 16 bit co-processor port
- Built-in self diagnostic software
- Automatic expansion with AppleWorks 1.3 or later
- Allows Apple II+ and IIe to run your AppleWorks without buying additional software
- Accelerates AppleWorks
- Displays time and date on the AppleWorks screen with any ProDos clock
- Fits any I/O slot except slot 3
- Fully socketed and user upgradeable
- Much, much more

RamFactor with 256K	\$239
RamFactor with 512K	\$289
RamFactor with 1 MEG	\$389
RamFactor with 2-16 MEG	CALL
Battery Back-up Option	\$179
65C816 16 Bit Card	\$159

Order RamFactor today... with 15 day money back guarantee and our "no hassle" five year warranty. *Call* 9 a.m. to 11 p.m., 7 days, or *send* check or money order to Applied Engineering, MasterCard, Visa and C.O.D. welcome. Texas residents add 5% sales tax. Add \$10.00 if outside U.S.A.

AE Applied Engineering™
The Apple enhancement experts
(214) 241-6060

P.O. Box 798, Carrollton, TX 75006

it will be the same as the previous root,
or it will be the previous root + 1.

if it is the same as the previous root,
the argument must still be less than
the "next perfect square"

if it is prev. root + 1,
the argument must be greater than or
equal to the "next perfect square"

I keep a running value for "next perfect square". I start with 1 (the next perfect square after $0*0$ is $1*1$). Then each time I find that the argument has reached the value of the "next perfect square", I bump it up by adding $2*root+1$. Remember that $(n+1)^2 = n^2 + 2*n + 1$.

Lines 1550, 1780, and 1860 indicate visually that the program is running, and helped me find a bug or two. Lines 1930-2050 print out the important information when an error is detected.

Lines 1280-1320 allowed me to call SQRT from inside an Applesoft program. However, this is not foolproof because there may be page-zero conflicts as the program is now written. It worked fine for my tests, though.

```

1000 *SAVE S.PUTNEY-RBSC FISQR
1010 *-----
1020 *          ULTRA FAST INTEGER SQUARE ROOTS
1030 *
1040 *   BY: CHARLES H. PUTNEY
1050 *       18 QUINNS ROAD
1060 *       SHANKILL, CO. DUBLIN, IRELAND
1070 *
1080 * INPUT: X = ARG HIGH BYTE
1090 *         A = ARG LOW BYTE
1100 *
1110 * OUTPUT: Y=INTEGER SQUARE ROOT OF X,A
1120 *         X AND A DESTROYED
1130 *
1140 *-----
00- 1150 BAS.ARG      .EQ $00,01
02- 1160 NUMBER      .EQ $02,03
04- 1170 ARGSAV      .EQ $04,05
06- 1180 ARGLO       .EQ $06
07- 1190 TEN.TIMES   .EQ $07
08- 1200 OLD.ROOT    .EQ $08
09- 1210 NEW.ROOT    .EQ $09
0A- 1220 RR          .EQ $0A,0B
0C- 1230 SS          .EQ $0C,0D,0E
1240 *-----
1250 *          .OR $6000   OUT OF THE WAY
1260 *          .TF SQUARE ROOT.OBJ
1270 *-----
6000- A5 00 1280 BASENT LDA BAS.ARG  USE LOW = 0
6002- A6 01 1290        LDY BAS.ARG+1 HIGH = 1
6004- 20 C1 60 1300        JSR SQRT   TEST IT
6007- 84 00 1310        STY BAS.ARG  RETURN IN 0
6009- 60 1320        RTS
1330 *-----
600A- A9 00 1340 TEST
600C- 85 02 1350        LDA #0
600E- 85 03 1360        STA NUMBER
6010- 85 08 1370        STA NUMBER+1
6012- 85 0A 1380        STA OLD.ROOT
6014- 85 0B 1390        STA RR
6016- 85 0D 1400        STA RR+1
1410        STA SS+1

```

```

6018- 85 OE      1420      STA SS+2
601A- A9 01      1430      LDA #1
601C- 85 0C      1440      STA SS
601E- A5 02      1450      LDA NUMBER      SET UP FOR SQRT ENTRY
6020- A6 03      1460      LDX NUMBER+1
6022- 20 C1 60    1470      JSR SQRT      FIND THE SQUARE ROOT
6025- 84 09      1480      STY NEW.ROOT
6027- C4 08      1490      CPY OLD.ROOT
6029- F0 29      1500      BEQ .2      SAME AS OLD ROOT
602B- E6 08      1510      INC OLD.ROOT
602D- F0 4C      1520      BEQ .99      ERROR
602F- C4 08      1530      CPY OLD.ROOT
6031- D0 48      1540      BNE .99      ERROR
6033- EE F4 07    1550      INC $7F4
6036- A5 0C      1560      LDA SS      SS = RR
6038- 85 0A      1570      STA RR
603A- A5 0D      1580      LDA SS+1
603C- 85 0B      1590      STA RR+1
603E- 38        1600      SEC      SS = SS + R + R + 1
603F- 26 09      1610      ROL NEW.ROOT
6041- A9 00      1620      LDA #0
6043- 2A        1630      ROL
6044- 48        1640      PHA      SAVE HIBYTE OF 2*R+1
6045- A5 0C      1650      LDA SS
6047- 65 09      1660      ADC NEW.ROOT
6049- 85 0C      1670      STA SS
604B- 68        1680      PLA
604C- 65 0D      1690      ADC SS+1
604E- 85 0D      1700      STA SS+1
6050- 90 02      1710      BCC .2
6052- E6 0E      1720      INC SS+2
6054- A5 02      1730      LDA NUMBER      ERROR IF NUMBER < RR
6056- C5 0A      1740      CMP RR
6058- A5 03      1750      LDA NUMBER+1
605A- E5 0B      1760      SBC RR+1
605C- 90 1D      1770      BCC .99
605E- EE F5 07    1780      INC $7F5
6061- A5 02      1790      LDA NUMBER      ERROR IF NUMBER >= SS
6063- C5 0C      1800      CMP SS
6065- A5 03      1810      LDA NUMBER+1
6067- E5 0D      1820      SBC SS+1
6069- A9 00      1830      LDA #0
606B- E5 0E      1840      SBC SS+2
606D- B0 0C      1850      BCS .99
606F- EE F6 07    1860      INC $7F6
6072- E6 02      1870      INC NUMBER
6074- D0 A8      1880      BNE .1      WRAPPED ?
6076- E6 03      1890      INC NUMBER+1
6078- D0 A4      1900      BNE .1      DONE 65536 ?
607A- 60        1910      RTS
607B- A5 03      1920      *-----
607D- 20 DA FD    1930      .99 LDA NUMBER+1
6080- A5 02      1940      JSR $FDDA
6082- 20 DA FD    1950      LDA NUMBER
6085- A9 AD      1960      JSR $FDDA
6087- 20 ED FD    1970      LDA #"-"
608A- A5 08      1980      JSR $FDED
608C- 20 DA FD    1990      LDA OLD.ROOT
608F- A9 AD      2000      JSR $FDDA
6091- 20 ED FD    2010      LDA #"-"
6094- A5 09      2020      JSR $FDED
6096- 20 DA FD    2030      LDA NEW.ROOT
6099- 60        2040      JSR $FDDA
6099- 60        2050      RTS
6099- 60        2060      *-----
6099- 60        2070      TIMING
609A- A9 00      2080      LDA #$00      SET UP NUMBER FOR INCREMENTING
609C- 85 02      2090      STA NUMBER
609E- 85 03      2100      STA NUMBER+1
60A0- A9 0A      2110      LDA #10      DO IT ALL TEN TIMES
60A2- 85 07      2120      STA TEN.TIMES
60A4- AD 59 CO    2130      LDA $C059      START TIMER
60A7- A5 02      2140      .1 LDA NUMBER      SET UP FOR SQRT ENTRY
60A9- A6 03      2150      LDX NUMBER+1
60AB- 20 C1 60    2160      JSR SQRT      FIND THE SQUARE ROOT
60AB- 20 C1 60    2170      * LDA $C030      REMOVE "*" TO GET NEAT SOUNDS

```


7 M E G I l e / 1 M E G I l e

	MultiRam	MultiRam
	RGB	Ile
	Card	Card
64k MULTIRAM	169.	129.
128k MULTIRAM	179.	139.
320k MULTIRAM	199.	164.
576k MULTIRAM	229.	199.
832k MULTIRAM	254.	234.
1024k MULTIRAM	279. ←	← ← ←
1344k MULTIRAM	419.	-
1600k MULTIRAM	449.	-
1792k MULTIRAM	479.	-

Apple Ile Enhancement Kit	62.
Cermetek 1200 Baud internal modem Ile/II+	184.
Incomm 2400 Baud external univ. modem	359.
Clockworks Clock (Thunder/Time HO™ comp)	89.
13" Teknika RGB Monitor*	322.
Accelerator Ile-350% speedup card	227.
Pinpoint Program or Spelling Checker (ea)	49.
ProAPP 10 & 20 Meg Hard Disk Drives	Call
Pico™ Slimline Drive Ile/II+	158.
256k Memory Chips-1 yr warranty (8)	42.

Checkmate Technology's MultiRam CX card easily expands your Ile to 640k and has a CX+ Piggyback to add another 512k (over 1.1 Meg total)! It's 100% compatible with all Ile software/hardware & unlike Z-Ram™ & other Ile cards, can be upgraded with 65C816 kit (\$129) to likely run new technology software! It comes with the **SAME FREE SOFTWARE, UPDATES & 5 YR WARRANTY AS Ile CARDS**

	OUR LOWEST PRICE
256k MULTIRAM CX	189.
512k MULTIRAM CX	222.
512k MULTIRAM CX+	CALL
1200 Baud Ile Modem (+ \$20/cable)	159.
Clip-on Ile Rechargeable Battery	129.
Ile Flat Panel Display	379.
Ile System Clock (Same as A.E.)	66.

Terms: Add \$4-Ground or \$6-Air shipping & phone # to each U.S. card order (foreign orders/FPO/APO extra). Add 3% for MasterCard/Visa (include #/expir) & P.O.'s. For fast delivery send Cashier's/Certified check, Money Order, C.O.D. (add \$6) & personal checks accepted (allow 16 days). **Tax res add 6 1/2% tax.**

MultiRam, Timemaster II H/O/Z Ram, Pico, respective trademarks of Checkmate Technology, Applied Engineering, WGE.

MCI Mail 2969684 • Telex 6502969684 (WUI)

WHY BUY FROM US RATHER THAN SOME MAIL ORDER HOUSES? Only we offer an exclusive 15 day MultiRam money back satisfaction guarantee, double software, more support, **FREE AUTOMATIC** software updates, **FREE 64k** with each 256k/512k/768k Ile card. We know the products well, & we have them in stock. **CALL FOR DETAILS, PRICES, QUANTITY DISCOUNTS. SCHOOLS & GROUPS WELCOME!**

COIT VALLEY COMPUTERS • 14055 Waterfall Way, Dallas, TX 75240 • (214) 234-5047

```

60AE- E6 02 2180 INC NUMBER
60B0- D0 F5 2190 BNE .1 WRAPPED ?
60B2- E6 03 2200 INC NUMBER+1
60B4- D0 F1 2210 BNE .1 DONE 65536 ?
60B6- EE F7 07 2220 INC $7F7
60B9- C6 07 2230 DEC TEN.TIMES
60BB- D0 EA 2240 BNE .1
60BD- AD 58 C0 2250 LDA $C058 STOP TIMER
60C0- 60 2260 RTS
2270 *-----
2280 SQR T
60C1- E0 40 2290 CPX #$40 VALUE ALREADY NORMALIZED?
60C3- 90 14 2300 BCC .2 .NO
2310 *---ARG = $4000...FFFF-----49152 CASES
60C5- E0 FF 2320 CPX #$FF CHECK FOR ARG-HI = $FF
60C7- F0 51 2330 BEQ .9 ...YES, SPECIAL CASE
60C9- BC 00 62 2340 LDY ROOT,X GET ROOT, USE AS INDEX
60CC- D9 00 63 2350 CMP TABLE2,Y
60CF- 90 07 2360 BCC .1 ...SPEEDS UP AVERAGE BY 0.8 CYCLE
60D1- 8A 2370 TXA ARG-HI
60D2- F9 00 64 2380 SBC TABLE3,Y
60D5- 90 01 2390 BCC .1
60D7- C8 2400 INY
60D8- 60 2410 .1 RTS
2420 *---ARG = $0000...3FFF-----
60D9- 86 05 2430 .2 STX ARGSAV+1 SAVE ARG-HI
60DB- E0 00 2440 CPX #0 IS ARG-HI ZERO?
60DD- F0 2B 2450 BEQ .7 .YES
2460 *---ARG = $01FF...3FFF-----16128 CASES
60DF- 85 04 2470 STA ARGSAV SAVE ARG-LO FOR SHIFTING
60E1- 85 06 2480 STA ARGLO SAVE ARG-LO FOR LATER COMPARE
60E3- 8A 2490 TXA ARG-HI TO A-REG
60E4- A0 00 2500 LDY #0 START SHIFT COUNT = 0
60E6- 06 06 2510 .3 ASL ARGLO
60E8- 2A 2520 ROL
60E9- 06 06 2530 ASL ARGLO
60EB- 2A 2540 ROL
60EC- C8 2550 INY
60ED- C9 40 2560 CMP #$40
60EF- 90 F5 2570 BCC .3
2580 *---A=NORM-ARG, Y=SHIFT-CNT-----
60F1- AA 2590 .4 TAX USE NORM-ARG FOR INDEX
60F2- BD 00 62 2600 LDA ROOT,X GET ROOT FROM TABLE
60F5- 4A 2610 .5 LSR HALF ROOT SHIFT-CNT TIMES
60F6- 88 2620 DEY
60F7- D0 FC 2630 BNE .5
60F9- A8 2640 TAY USE SHIFTED ROOT FOR INDEX NOW
60FA- A5 04 2650 LDA ARGSAV GET ARG-LO
60FC- D9 00 63 2660 CMP TABLE2,Y
60FF- 90 08 2670 BCC .6 ...SPEEDS UP AVERAGE BY 0.7 CYCLE
6101- A5 05 2680 LDA ARGSAV+1
6103- F9 00 64 2690 SBC TABLE3,Y
6106- 90 01 2700 BCC .6
6108- C8 2710 INY
6109- 60 2720 .6 RTS
2730 *---ARG = $0000...00FF-----
610A- A8 2740 .7 TAY IS ARG-LO ALSO ZERO?
610B- F0 CB 2750 BEQ .1 ...YES, SQR T=0
2760 *---ARG = $0001. 00FF-----255 CASES
610D- 85 04 2770 STA ARGSAV SAVE ARG-LO FOR LATER COMPARE
610F- A0 04 2780 LDY #4 START SHIFT COUNT = 4
6111- C9 40 2790 .8 CMP #$40 NORMALIZED YET?
6113- B0 DC 2800 BCS .4 ...YES, GET ROOT NOW
6115- 0A 2810 ASL
6116- 0A 2820 ASL
6117- C8 2830 INY COUNT THE SHIFT
6118- D0 F7 2840 BNE .8 .ALWAYS
2850 *---ARG = $FFXX-----
611A- A0 FF 2860 .9 LDY #$FF
611C- 60 2870 RTS
2880 *-----
5C- 2890 ZZ .EQ *-SQR T
2900 *-----
611D- 2910 * PUT TABLES SO NO PAGE CROSSING
2920 .BS *+255/256*256-*+64
2930 *-----

```

```

2940 *      DON'T WASTE PAPER
2950 .LIST MOFF
2960 .MA HS
2970 .HS J1
2980 .EM
2990 *-----
3000 *      SQUARE ROOT TABLE OF N
3010 *      FROM $4000 (16384)
3020 *      TO $FF00 (65280)
3030 *      BY $100 (256)
3040 *
6240- 3040 TABLE1 >HS 80.80.81.82.83.84.85.86.
6248- 3050 >HS 87.88.89.8A.8B.8C.8D.8E.
6250- 3060 >HS 8F.90.90.91.92.93.94.95.
6258- 3070 >HS 96.96.97.98.99.9A.9B.9B.
6260- 3080 >HS 9C.9D.9E.9F.A0.A0.A1.A2.
6268- 3090 >HS A3.A3.A4.A5.A6.A7.A8.
6270- 3100 >HS A9.AA.AA.AB.AC.AD.AE.
6278- 3110 >HS AF.B0.B0.B1.B2.B3.B4.
6280- 3120 >HS B5.B5.B6.B7.B7.B8.B9.B9.
6288- 3130 >HS BA.BB.BB.BC.BD.BD.BE.BF.
6290- 3140 >HS C0.C0.C1.C1.C2.C3.C4.
6298- 3150 >HS C5.C5.C6.C7.C7.C8.C9.
62A0- 3160 >HS CA.CB.CB.CC.CD.CE.CE.
62A8- 3170 >HS CF.D0.D0.D1.D1.D2.D3.
62B0- 3180 >HS D4.D4.D5.D6.D6.D7.D8.
62B8- 3190 >HS D9.D9.DA.DA.DB.DB.DC.
62C0- 3200 >HS DD.DE.DE.DF.E0.E0.E1.E1.
62C8- 3210 >HS E2.E2.E3.E3.E4.E5.E6.
62D0- 3220 >HS E6.E7.E7.E8.E8.E9.EA.
62D8- 3230 >HS EB.EB.EC.EC.ED.ED.EE.
62E0- 3240 >HS EF.F0.F0.F1.F1.F2.F3.
62E8- 3250 >HS F3.F4.F4.F5.F5.F6.F7.
62F0- 3260 >HS F7.F8.F8.F9.F9.FA.FB.
62F8- 3270 >HS FB.FC.FC.FD.FD.FE.FF.
3280 *-----
3290 *
3300 *      SQUARE TABLE CONTAINING LOW
3310 *      BYTE OF (N+1)
3320 *
6300- 3320 TABLE2 >HS 01.04.09.10.19.24.31.40.
6308- 3330 >HS 51.64.79.90.A9.C4.E1.00.
6310- 3340 >HS 21.44.69.90.B9.E4.11.40.
6318- 3350 >HS 71.A4.D9.10.49.84.C1.00.
6320- 3360 >HS 41.84.C9.10.59.A4.F1.40.
6328- 3370 >HS 91.E4.39.90.E9.44.A1.00.
6330- 3380 >HS 61.C4.29.90.F9.64.D1.40.
6338- 3390 >HS B1.24.99.10.89.04.81.00.
6340- 3400 >HS 81.04.89.10.99.24.B1.40.
6348- 3410 >HS D1.64.F9.90.29.C4.61.00.
6350- 3420 >HS A1.44.E9.90.39.E4.91.40.
6358- 3430 >HS F1.A4.59.10.C9.84.41.00.
6360- 3440 >HS C1.84.49.10.D9.A4.71.40.
6368- 3450 >HS 11.E4.B9.90.69.44.21.00.
6370- 3460 >HS E1.C4.A9.90.79.64.51.40.
6378- 3470 >HS 31.24.19.10.09.04.01.00.
6380- 3480 >HS 01.04.09.10.19.24.31.40.
6388- 3490 >HS 51.64.79.90.A9.C4.E1.00.
6390- 3500 >HS 21.44.69.90.B9.E4.11.40.
6398- 3510 >HS 71.A4.D9.10.49.84.C1.00.
63A0- 3520 >HS 41.84.C9.10.59.A4.F1.40.
63A8- 3530 >HS 91.E4.39.90.E9.44.A1.00.
63B0- 3540 >HS 61.C4.29.90.F9.64.D1.40.
63B8- 3550 >HS B1.24.99.10.89.04.81.00.
63C0- 3560 >HS 81.04.89.10.99.24.B1.40.
63C8- 3570 >HS D1.64.F9.90.29.C4.61.00.
63D0- 3580 >HS A1.44.E9.90.39.E4.91.40.
63D8- 3590 >HS F1.A4.59.10.C9.84.41.00.
63E0- 3600 >HS C1.84.49.10.D9.A4.71.40.
63E8- 3610 >HS 11.E4.B9.90.69.44.21.00.
63F0- 3620 >HS E1.C4.A9.90.79.64.51.40.
63F8- 3630 >HS 31.24.19.10.09.04.01.00.
3640 *-----
3650 *
3660 *      SQUARE TABLE CONTAINING HIGH
3670 *      BYTE OF (N+1)
3680 *
6400- 3680 TABLE3 >HS 00.00.00.00.00.00.00.00.
6408- 3690 >HS 00.00.00.00.00.00.00.01.

```

6410-	3700	>HS 01.01.01.01.01.01.02.02.
6418-	3710	>HS 02.02.02.03.03.03.03.04.
6420-	3720	>HS 04.04.04.05.05.05.05.06.
6428-	3730	>HS 06.06.07.07.07.08.08.09.
6430-	3740	>HS 09.09.0A.0A.0A.0B.0B.0C.
6438-	3750	>HS 0C.0D.0D.0E.0E.0F.0F.10.
6440-	3760	>HS 10.11.11.12.12.13.13.14.
6448-	3770	>HS 14.15.15.16.17.17.18.19.
6450-	3780	>HS 19.1A.1A.1B.1C.1C.1D.1E.
6458-	3790	>HS 1E.1F.20.21.21.22.23.24.
6460-	3800	>HS 24.25.26.27.27.28.29.2A.
6468-	3810	>HS 2B.2B.2C.2D.2E.2F.30.31.
6470-	3820	>HS 31.32.33.34.35.36.37.38.
6478-	3830	>HS 39.3A.3B.3C.3D.3E.3F.40.
6480-	3840	>HS 41.42.43.44.45.46.47.48.
6488-	3850	>HS 49.4A.4B.4C.4D.4E.4F.51.
6490-	3860	>HS 52.53.54.55.56.57.59.5A.
6498-	3870	>HS 5B.5C.5D.5F.60.61.62.64.
64A0-	3880	>HS 65.66.67.69.6A.6B.6C.6E.
64A8-	3890	>HS 6F.70.72.73.74.76.77.79.
64B0-	3900	>HS 7A.7B.7D.7E.7F.81.82.84.
64B8-	3910	>HS 85.87.88.8A.8B.8D.8E.90.
64C0-	3920	>HS 91.93.94.96.97.99.9A.9C.
64C8-	3930	>HS 9D.9F.A0.A2.A4.A5.A7.A9.
64D0-	3940	>HS AA.AC.AD.AF.B1.B2.B4.B6.
64D8-	3950	>HS B7.B9.BB.BD.BE.C0.C2.C4.
64E0-	3960	>HS C5.C7.C9.CB.CC.CE.D0.D2.
64E8-	3970	>HS D4.D5.D7.D9.DB.DD.DF.E1.
64F0-	3980	>HS E2.E4.E6.E8.EA.EC.EE.F0.
64F8-	3990	>HS F2.F4.F6.F8.FA.FC.FE.F0.
	4000	-----
6200-	4010	* ROOT .EQ TABLE1-\$40 SET UP SO \$80 IS FIRST SQUARE ROOT
6300-	4020	EXACTL .EQ TABLE2 SET UP SO 0 INDEX (OF \$4000)
6400-	4030	EXACTH .EQ TABLE3 GIVES EXACT SQUARE OF 1
	4040	-----

We Make Measurement And Control Easy!

12 BIT, 16 CHANNEL, PROGRAMMABLE GAIN A/D

- All new 1984 design incorporates the latest in state-of-art I.C. technologies.
- Complete 12 bit A/D converter, with an accuracy of 0.02%!
- 16 single ended channels (single ended means that your signals are measured against the Apple's GND) or 8 differential channels. Most all the differential channels. Most all the single ended.
- 9 software programmable full scale ranges, any of the 16 channels can have any range at any time. Under program control, you can select any of the following ranges: ± 10 volts, ± 5 V, ± 1.5 V, ± 1.0 V, ± 500 mV, ± 250 mV, ± 100 mV, ± 50 mV, or ± 25 mV.
- Very fast conversion (25 micro seconds).
- Analog input resistance greater than 1,000,000 ohms.
- Laser-trimmed scaling resistors.
- Low power consumption through the use of CMOS devices.
- The user connector has +12 and -12 volts on it so you can power your sensors.
- Only elementary programming is required to use the A/D.
- The entire system is on one standard size plug-in card that fits neatly inside the Apple.
- System includes sample programs on disk.

PRICE \$319

A few applications may include the monitoring of ● flow ● temperature ● humidity ● wind speed ● wind direction ● light intensity ● pressure ● RPM ● soil moisture and many more.

A/D & D/A

- A/D & D/A Features:
 - Single PC card
 - 8 channels A/D
 - 8 channels D/A
 - Superfast conversion time
 - Very easy programming
 - Many analog ranges
 - Manual contains sample applications

A/D SPECIFICATIONS

- 0.3% accuracy
 - On-board memory
 - Fast conversion (0.78 MS per channel)
 - A/D process totally transparent to Apple (looks like memory)
 - User programmable input ranges are 0 to 10 volts, 0 to 5, -5 to +5, -2.5 to +2.5, -5 to 0, -10 to 0.
- The A/D process takes place on a continuous, channel sequencing basis. Data is automatically transferred to its proper location in the on-board RAM. No A/D converter could be easier to use.

D/A SPECIFICATIONS

- 0.3% accuracy
 - On-board memory
 - On-board output buffer amps can drive 5 MA
 - D/A process is totally transparent to the Apple (just poke the data)
 - Fast conversion (0.03 MS per channel)
 - User programmable output ranges are 0 to 5 volts and 0 to 10 volts
- The D/A section contains 8 digital to analog converters, with output buffer amplifiers and all interface logic on a single card. On-card latches are provided for each of the eight D/A converters. No D/A converter could be easier to use. The on-board amplifiers are laser-trimmed during manufacture, thereby eliminating any requirement for off-set nulling.

PRICE \$199

SIGNAL CONDITIONER

Our 8 channel signal conditioner is designed for use with both our A/D converters. This board incorporates 8 F.E.T. op-amps, which allow almost any gain or offset. For example, an input signal that varies from 2.00 to 2.15 volts or a signal that varies from 0 to 50 mV can easily be converted to 0-10V output for the A/D.

The signal conditioner's outputs are on a high quality 16 pin gold I.C. socket that matches the one on the A/D's so a simple ribbon cable connects the two. The signal conditioner can be powered by your Apple or from an external supply.

FEATURES

- 4 1/2" square for standard card cage and 4 mounting holes for standard mounting. The signal conditioner does not plug into the Apple, it can be located up to 1/2 mile away from the A/D.
- 22 pin 156 spacing edge card input connector (extra connectors are easily available i.e. Radio Shack)
- Large board area
- Full detailed schematic included.

PRICE \$79

I/O 32

- Provides 4, 8-Bit programmable I/O Ports
- Your inputs can be anything from high speed logic to simple switches
- Any of the 4 ports can be programmed as an input or an output port
- Programming is made very easy by powerful on-board firmware
- All I/O lines are TTL (0-5 volt) compatible
- The I/O 32 is your best choice for any control application

The I/O manual includes many programs for inputs and outputs.

Some applications include:

Burglar alarm, direction sensing, use with relays to turn on lights, sound buzzers, start motors, control tape recorders and printers, use with digital joystick.

PRICE \$89

Please see our other full page ad in this magazine for information on Applied Engineering's Timemaster Clock Card and other products for the Apple.

Our boards are far superior to most of the consumer electronics made today. All I.C.'s are in high quality sockets with mil-spec. components used throughout. P.C. boards are glass-epoxy with gold contacts. Made in America to be the best in the world. All products compatible with Apple II and IIe.

Applied Engineering's products are fully tested with complete documentation and available for immediate delivery. All products are guaranteed with a no hassle three year warranty.

Texas Residents Add 5% Sales Tax
Add \$10.00 if Outside U.S.A.

Send Check or Money Order to:
APPLIED ENGINEERING
P.O. Box 798
Carrollton, TX 75006

Call (214) 241-4060
9 a.m. to 11 p.m. 7 days a week
MasterCard, Visa & C.O.D. Welcome
No extra charge for credit cards

New ProDOS Bug and Fix

The November 1986 issue of Open-Apple (Tom Weishaar's wonderful newsletter) tells of an important new discovery. For about a year Tom has been reporting on the symptom: Appleworks and Applewriter data disks suddenly turning up with track 0 destroyed. It only happened to 5.25" diskettes, and only one certain machines, and otherwise seemingly at random. For a complete description, get all of Tom's back issues.

Some of his readers from Australia seem to have tracked down the problem, and they suggest a solution. In the floppy driver code inside ProDOS, at \$D6C3, there are four STA commands that turn off all four stepper motor windings. Tom says the purpose is to disable any 3.5" drives connected in a daisy chain to the same controller. I wonder, because this code has been here since 1983, long before the possibility of 3.5" drives. Anyway, the code has a bad side-effect in some systems.

A quirk of the controller card is that STA operations to the stepper motor winding soft-switches also cause the card to write on the data bus. So you have the bus being driven in two directions at once: the cpu trying to store the A-register, and the controller card trying to send something meaningless. Besides resulting in garbage on the data bus, which causes no real damage in this case, apparently in some Apples with some controller cards it causes the card to go into WRITE mode. Whatever track the head is sitting on will then be clobbered.

The solution is to change the four STA operations to LDA. The disk drives will get the same message, without causing the bus contention. You can patch the PRODOS system file and re-SAVE it, on all your disks. If you have a hard disk, you should only have to do it one time. If you BLOAD the PRODOS file at \$2000, the four instructions will be found at \$56D3:

```
56D3: 9D 80 C0  STA $C080,X
56D6: 9D 82 C0  STA $C082,X
56D9: 9D 84 C0  STA $C084,X
56DC: 9D 86 C0  STA $C086,X
```

If you change all those "9D" bytes to "BD", which is the opcode for "LDA addr,X", the bug is supposed to disappear. Doing it from inside the S-C Macro Assembler, I did it this way:

```
:BLOAD PRODOS,TSYS,A$2000
:UNLOCK PRODOS
:$56D3:BD N 56D6:BD N 56D9:BD N 56DC:BD
:BSAVE PRODOS,TSYS,A$2000,L14848
:LOCK PRODOS
```

I personally have never had ProDOS clobber a diskette. I have trashed some myself, by stupidity, but this hardware/software bug has never caused it. Nevertheless, I have now patched my disks, just in case. Many thanks to Tom, Open-Apple, and to the men in Australia.



NEW !!!][IN A MAC: \$69.00

This Apple II emulator runs DOS 3.3 and PRODOS programs (including 6502 machine language routines) on a 512K Macintosh. All Apple II features are supported such as HI-RES/LO-RES graphics, 40/80 column text screens, language card and joystick. Also included: clock, RAM disk, keyboard buffer, on-screen HELP, access to the desk accessories and support for 4 logical disk drives. Package includes 2 MAC diskettes (PROGRAM holds emulation, communications and utility software, DATA holds DOS 3.3 and PRODOS system masters, including Applesoft and Integer BASIC) and 1 Apple II diskette (transfer software moves disk images to the MAC).

NEW !!! SCREEN.GEN: \$35.00

Develop HI-RES screens for your Apple II on a Macintosh. Don't be limited by MousePaint or other screen editors. Use MACPAINT (or any other application) on the MAC to create your Apple II screen. Then use SCREEN.GEN to transfer directly from the MAC to the Apple II (with SuperSerial card or equivalent). Package includes Apple II diskette with transfer software plus fully commented SOURCE code.

NEW !!! MIDI-MAGIC for Apple //c: \$49.00

Compatible with any MIDI equipped music keyboard, synthesizer, organ or piano. Package includes a MIDI-out cable (plugs directly into modem port - no modifications required!) and 6-song demo diskette. Large selection of digitized QRS player-piano music available for 19.00 per diskette (write for catalog). MIDI-MAGIC compatible with Apple II family using Passport MIDI card (or our own input/output card w/drum sync for only \$99.00).

FONT DOWNLOADER & EDITOR: \$39.00

Turn your printer into a custom typesetter. Downloaded characters remain active while printer is powered. Use with any Word Processor program capable of sending ESC and control codes to printer. Switch back and forth easily between standard and custom fonts. Special printer functions (like expanded, compressed etc.) supported. HIRES screen editor lets you create your own characters and special graphics symbols. For Apple II, II+, //e. Specify printer: Apple Dot Matrix, C.Itoh 8510A (Prowriter), Epson FX 80/100, or OkiData 92/93.

* The Font Downloader & Editor for the Apple Imagewriter Printer. For use with Apple II, II+, //e (with SuperSerial card) and the Apple //c (with builtin serial interface).

* FONT LIBRARY DISKETTE #1: \$19.00 contains lots of user-contributed fonts for all printers supported by the Font Downloader & Editor. Specify printer with order.

DISASM 2.2e : \$30.00 (\$50.00 with SOURCE Code)

Use this intelligent disassembler to investigate the inner workings of Apple II machine language programs. DISASM converts machine code into meaningful, symbolic source compatible with S-C, LISA, Toolkit and other assemblers. Handles data tables, displaced object code & even provides label substitution. Address-based triple cross reference generator included. DISASM is an invaluable machine language learning aid to both novice & expert alike. Don Lancaster says DISASM is "absolutely essential" in his ASSEMBLY COOKBOOK.

The 'PERFORMER' CARD: \$39.00 (\$59.00 with SOURCE Code)

Converts a 'dumb' parallel printer I/F card into a 'smart' one. Command menu eliminates need to remember complicated ESC codes. Features include perforation skip, auto page numbering with date & title. Includes large HIRES graphics & text screen dumps. Specify printer: MX-80 with Grafix-80, MX-100, MX-80/100 with Grafixplus, NEC 8092A, C.Itoh 8510 (Prowriter), OkiData 82A/83A with Okigraph & OkiData 92/93.

'MIRROR' ROM: \$25.00 (\$45.00 with SOURCE Code)

Communications ROM plugs directly into Novation's Apple-Cat Modem card. Basic modes: Dumb Terminal, Remote Console & Programmable Modem. Features include: selectable pulse or tone dialing, true dialtone detection, audible ring detect, ring-back, printer buffer, 80 col card & shift key mod support.

RAM/ROM DEVELOPMENT BOARD: \$30.00

Plugs into any Apple slot. Holds one user-supplied 2Kx8 memory chip (6116 type RAM for program development or 2716 EPROM to keep your favorite routines on-line). Maps into \$Cn00-CnFF and \$C800-CFFF.

C-PRINT For The APPLE //c: \$69.00

Connect standard parallel printers to an Apple //c. C-PRINT plugs into the standard Apple //c printer serial port and into any printer having a standard 36 pin centronics-type parallel connector. Just plug in and print!

Unless otherwise specified, all Apple II diskettes are standard (not copy protected!) 3.3 DOS.

Avoid a \$3.00 handling charge by enclosing full payment with order. VISA/MC and COD phone orders OK.

RAK-WARE 41 Ralph Road W. Orange N.J. 07052 (201) 325-1885



Timing Apple Programs with Another Apple.....

...Bob Sander-Cederlof

While I was working on Charles Putney's integer square root program, I longed for a better way to time it. I was wasting a lot of my time using a stopwatch, and still getting inaccurate (or at least imprecise) times.

For around \$3000 I could buy a logic analyzer and hook it up to count machine cycles. That is obviously out of the question. Maybe I could hunt around among my old boards and find one with a 6522 on it: that chip has an interval timer that could give me fairly accurate times. I might be able to find one, but then I would have to figure out how to program it again.

Then I thought about using the game port to communicate with another Apple, and put a timing loop in the other Apple. I hooked one of the Annunciator output lines in my first Apple to a Push Button input line on the second one. Then I set up the program being clocked to set the annunciator on at the beginning and turn it off at the end. I wrote a timing program to run in the other Apple which waited until the push button input went on, and then counted loops until it went low again. The results were better than I hoped for!

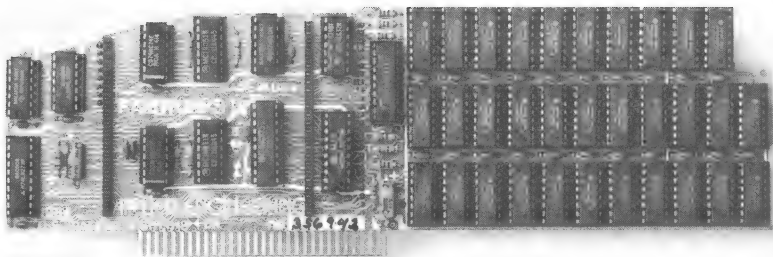
To hook up the Apples, I started by finding some wire. I needed about 12 feet of at least two wires. I found about six feet of four-line telephone wire, and another six feet of twisted pair left over from my burglar alarm installation. I connect them together, very crudely, and stretched them across the room. The Apple on the south side of the room is my nine-year-old. It has a nice ZIF-socket in the game port, so I inserted the ground wire into pin 8 and the signal wire into pin 2, and clamped the socket. If you do not have a ZIF socket in yours, the telephone wire fits very nicely into the holes in a regular socket.

The Apple //e on the north side of the room challenged me a little more. First, the game socket is unreachable, way under the top right lip of the upper case. I can't even see it without a flashlight! There is a nine-pin D-connector on the back panel, but the Annunciator lines do not come to this connector. A little research led to the knowledge that the Annunciator signals come directly from pins 10-13 of the IOU chip. I chose AN0, which is pin 10. I hooked a red miniclip lead to that pin, and a black miniclip lead to ground at pin 1 of the same chip. The IOU chip is the 40-pin chip at position E5 on my //e motherboard, conveniently labeled "IOU". Facing the computer from the front, pin one is the first one on the right-hand side of the chip. Pin 10 is on the same side, about half way back. I then connected the other end of those leads to my wires, and the circuit was complete.

The program in the //e is the program whose time I want to measure. At the beginning of the section to be timed, I insert the instruction "LDA \$C059" to turn on AN0. At the end, I insert the instruction "LDA \$C058" to turn off AN0.

RamWorks® III

Patented Performance from the Recognized Leader



With battery backed RAM port, RGB port, increased memory capacity, full software compatibility and more compact design, RamWorks III is a generation ahead.

RamWorks III is the newest 3rd generation RAM card for the Apple IIe. It incorporates all of the technology and improvements that years of experience and over a hundred thousand sales have given us. By selling more memory cards than anyone else and listening to our customers, we were able to design a memory card that has the ultimate in performance, quality, compatibility and ease of use. A design so advanced it's patented. We call it RamWorks III, you'll call it awesome!

The AppleWorks Amplifier.

While RamWorks III is recognized by all memory intensive programs, NO other expansion card comes close to offering the multitude of enhancements to AppleWorks that RamWorks III does. Naturally, you'd expect RamWorks III to expand the available desktop, after all Applied Engineering was a year ahead of everyone else *including Apple* in offering more than 55K, and we still provide the largest AppleWorks desktops available. But a larger desktop is just part of the story. Look at all the AppleWorks enhancements that even Apple's own card does not provide and *only* RamWorks III does. With a 256K or larger RamWorks III, *all* of AppleWorks (including printer routines) will automatically load itself into RAM dramatically increasing speed by eliminating the time required to access the program disk drive. Switch from word processing to spreadsheet to database at the speed of light with no wear on disk drives.

Only RamWorks eliminates AppleWorks' internal memory limits, increasing the maximum number of records available from 1,350 to over 25,000. *Only* RamWorks increases the number of lines permitted in the word processing mode from 2,250 to over 15,000. And *only* RamWorks offers a built-in printer buffer, so you won't have to wait for your printer to stop before returning to AppleWorks. RamWorks even expands the clipboard. And auto segments large files so they can be saved on two or more disks. You can even have Pinpoint or MacroWorks and your favorite spelling checker in RAM for instant response.

RamWorks, nothing comes close to enhancing AppleWorks so much.

The Most Friendly, Most Compatible Card Available.

Using RamWorks III couldn't be easier because it's compatible with more off-the-shelf software than any other RAM card. Popular programs like AppleWorks, Pinpoint, Catalyst, MouseDesk, Howard-Sofi, FlashCalc, Pro-Filer, Managing Your Money, SuperCalc 3a, and MagiCalc to name a few (and *all* hardware add on's like Profile and Sider hard disks). RamWorks is even compatible with software written for Apple cards. But unlike other cards, RamWorks plugs into the IIe auxiliary slot providing our super sharp 80 column text (U.S. Patent #4601081) in a completely integrated system while leaving expansion slots 1 through 7 available for other peripheral cards.

RamWorks III is compatible with all

Apple IIe's, enhanced, unenhanced, American or European versions.

Highest Memory Expansion.

Applied Engineering has always offered the largest memory for the IIe and RamWorks III continues that tradition by expanding to 1 full MEG on the main card using standard RAMs, more than most will ever need (1 meg is about 500 pages of text)...but if you do ever need more than 1 MEG, RamWorks III has the widest selection of expander cards available. Additional 512K, 2 MEG, or 16 MEG cards just snap directly onto RamWorks III by plugging into the industry's only low profile (no slot 1 interference) fully decoded memory expansion connector. You can also choose non-volatile, power independent expanders allowing permanent storage for up to 20 years.

It Even Corrects Mistakes.

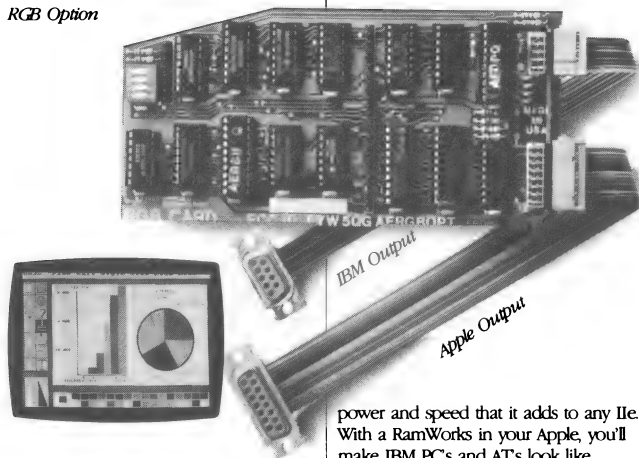
If you've got some other RAM card that's not being recognized by your programs, and you want RamWorks III, you're in luck. Because all you have to do is plug the memory chips from your current card into the expansion sockets on RamWorks to recapture most of your investment!

The Ultimate in RGB Color.

RGB color is an option on RamWorks and with good reason. Some others combine RGB color output with their memory cards, but that's unfair for those who don't need RGB *and* for those that do. Because if you don't need RGB

Applied Engineering doesn't make you buy it, but if you want RGB output you're in for a nice surprise because the RamWorks RGB option offers better color graphics plus a more readable 80 column text (that blows away any composite color monitor). For only \$129 it can be added to RamWorks giving you a razor sharp, vivid brilliance that most claim is the best they have ever seen. You'll also appreciate the multiple text colors (others only have green) that come standard. But the RamWorks RGB option is more than just the ultimate in color output because unlike others, it's fully compatible with all the Apple standards for RGB output control, making it more compatible with off-the-shelf software. With its FCC certified design, you can use almost any RGB monitor because only the new RamWorks RGB option provides both Apple standard and IBM standard RGB outputs (cables included). The RGB option plugs into the back of RamWorks with no slot 1 inter-

RGB Option



ference and remember you can order the RGB option with your RamWorks or add it on at a later date.

True 65C816 16 Bit Power.

RamWorks III has a built-in 65C816 CPU port for direct connection to our optional 65C816 card. The only one capable of linearly addressing more than 1 meg of memory for power applications like running the Lotus 1-2-3™ compatible program, VIP Professional. Our 65C816 card does not use another slot but replaces the 65C02 yet maintains full 8 bit compatibility.

Endorsed by the Experts.

A+ magazine said "Applied Engineering's RamWorks is a boon to those who must use large files with AppleWorks...I like the product so much that I am buying one for my own system." inCider magazine said "RamWorks is the most



Steve Wozniak, the creator of Apple Computer

"I wanted a memory card for my Apple that was fast, easy to use, and very compatible; so I bought RamWorks."

powerful auxiliary slot memory card available for your IIe, and I rate it four stars...For my money, Applied Engineering's RamWorks is king of the hill."

Apple experts everywhere are impressed by RamWorks's expandability, versatility, ease of use, and the sheer

power and speed that it adds to any IIe. With a RamWorks in your Apple, you'll make IBM PC's and AT's look like slowpokes.

Quality and Support of the Industry Leader

RamWorks III is from Applied Engineering, the largest, most well supported manufacturer of Apple peripherals and the inventor of large RAM cards for the Apple. With our 5 year no hassle warranty and outstanding technical support, you're assured of the most trouble free product you can buy.

It's Got It All

- 15 day money back guarantee
- 5 year hassle free warranty insures

coverage no matter where you purchase

- Built-in super sharp 80 column display, (U.S. Patent #4601081)
- Expandable to 1 MEG on main card
- Expandable to 16 meg with expander cards, with NO slot 1 interference
- Can use 64K or 256K RAMs
- Powerful linear addressing 16 bit coprocessor port
- Automatic AppleWorks expansion up to 3017K desktop
- Accelerates AppleWorks
- Built-in AppleWorks printer buffer
- The only large RAM card that's 100% compatible with all IIe software
- RamDrive™ the ultimate disk emulation software included free
- Memory is easily partitioned allowing many programs in memory at once
- Compatible, RGB option featuring ultra high resolution color graphics and multiple text colors, with cables for both Apple and IBM type monitors
- Built-in self diagnostics software
- Lowest power consumption (U.S. Patent #4601081)
- Takes only one slot (auxiliary) even when fully expanded
- Socketed and user upgradeable
- Software industry standard
- Advanced Computer Aided Design
- Used by Apple Computer, Steve Wozniak and virtually all software companies
- Displays date and time on the AppleWorks screen with any PRO-DOS compatible clock
- Much, much more!

RamWorks III with 64K	\$179
RamWorks III with 256K	\$219
RamWorks III with 512K	\$269
RamWorks III with 1 MEG	\$369
RamWorks III with 1.5 MEG	\$539
RamWorks III with 2 to 16 MEG	CALL
65C816 16 Bit Card	\$159
RGB Option	\$129
Optional Software:	
Pinpoint with RAM	
Enhancement Software	\$79
VIP Professional	\$219

RamWorks III. The industry standard for memory expansion of the Apple IIe.

ORDER YOUR RamWorks III TODAY. 9 a.m. to 11 p.m. 7 days, or send check or money order to Applied Engineering.

MasterCard, Visa and C.O.D. welcome. Texas residents add 5% sales tax. Add \$10.00 if outside U.S.A.

AE Applied Engineering™
P.O. Box 798, Carrollton, TX 75006
(214) 241-6060

The timing program in the other Apple is shown below. Lines 1130-1180 set up a page zero location to contain \$01, which I need later to make all the timing correct. They also clear the three registers, which I am going to use for accumulating a 24-bit count. Lines 1190-1200 then wait until the input signal goes high. This will happen when the program in the //e does the "LDA \$C059" instruction.

Lines 1260-1400 increment the 24-bit count once each 20 cycles, until the PBO signal falls. The signal is tested only once each 20 cycles, so there is a built in resolution of 20 cycles. If I want to measure a program down to the exact cycle, I will have to run it at least 20 times. Actually, there are two other sources of "error": the signal on my 12 feet of wire will not necessarily rise and fall at exactly the same speed; and the two Apples may not be running at exactly the same speed.

The various paths in lines 1260-1400 are all carefully timed so they all take exactly 20 cycles. The interval between BIT PBO executions should always be 20 cycles. Of course, that is, unless I made a mistake. There is one exception: When the A-register wraps around, after 16,777,216 counts, lines 1340-1350 add 5 cycles. The total interval on this path is 24 cycles. But this only happens once every 6 or 7 minutes, so who cares!

Finally, lines 1420-1490 print out the resulting count in hexadecimal. I then take my handy Radio Shack calculator out, convert to decimal, multiply by 20, and have the cycle count.

I like this arrangement so well, and I need to time programs so frequently, that I plan to make a more permanent hookup. And that reminds me of an old idea... a way to network several Apples using just the gameport....

```

1000 *SAVE S.TIMER
1010 *-----
1020 *   START COUNT WHEN PBO (OPEN-APPLE)
1030 *   PRESSED, STOP WHEN RELEASED
1040 *-----
00- 1050 CNT0 .EQ 0
01- 1060 CNT1 .EQ 1
02- 1070 CNT2 .EQ 2
03- 1080 ONE .EQ 3
1090 *-----
C061- 1100 PBO .EQ $C061   BIT 7 = 1 WHEN PRESSED
1110 *-----
1120 T
0800- A0 01 1130 LDY #1
0802- 84 03 1140 STY ONE
0804- 88 1150 DEY Y=0
0805- 98 1160 TYA A=0
0806- AA 1170 TAX X=0
0807- 18 1180 CLC
0808- 2C 61 C0 1190 .1 BIT PBO
080B- 10 FB 1200 BPL .1
1210 *-----
1220 *   20 CYCLES PER LOOP, REGARDLESS OF PATH
1230 *   24-BIT COUNTER GIVES 16,777,216 COUNTS
1240 *   WHICH IS 335,544,320 CYCLES
1250 *-----
080D- 2C 61 C0 1260 .2 BIT PBO
0810- 10 12 1270 BPL .5   END OF COUNTING
0812- E8 1280 INX
0813- D0 0A 1290 BNE .3
0815- C8 1300 INY

```

0816-	D0 09	1310	BNE .4
0818-	65 03	1320	ADC ONE
081A-	D0 F1	1330	BNE .2
081C-	18	1340	CLC
081D-	90 EE	1350	BCC .2
		1360	*
081F-	EA	1370	.3 NOP
0820-	EA	1380	NOP
0821-	EA	1390	.4 NOP
0822-	D0 E9	1400	BNE .2
		1410	*-----
0824-	85 02	1420	.5 STA CNT2
0826-	84 01	1430	STY CNT1
0828-	86 00	1440	STX CNT0
082A-	20 DA FD	1450	JSR \$FDDA
082D-	A5 01	1460	LDA CNT1
082F-	20 DA FD	1470	JSR \$FDDA
0832-	A5 00	1480	LDA CNT0
0834-	4C DA FD	1490	JMP \$FDDA
		1500	*-----

The //gs Reference Manuals

When Apple sent me the prototype //gs they included 11 fat 3-ring binders full of documentation. Much of it is destined to eventually be published as reference manuals by Addison-Wesley. I can hardly wait, because in the present form it is incomplete, inconvenient, inconsistent, inaccurate, and takes up too much space. I am sure the finished product will be up to Apple's usual standard, eliminating all the negatives just mentioned.

Addison-Wesley has released a little folder which describes the new manuals, with projected publishing dates. We will carry some of these, as soon as they are available.

The first book out will be "Technical Introduction to the Apple //gs". It is due in December, but there is not much REAL information in it. It is more like a complete marketing description, without the kind of detailed information programmers need. It is only 120 pages.

Three books are due in "Spring, 1987". I suppose that means we can expect copies by June 21st, at least. These look like books worth ordering:

- "Programmer's Introduction to the Apple //gs",
150 pgs, \$19.95
- "Apple //gs Hardware Reference", 250 pgs, \$26.95
- "Apple //gs Firmware Reference", 250 pgs, \$24.95

Three more are due in "Summer, 1987", which means no later than September 21st:

- "Apple //gs Toolbox Reference"
Volume 1, 400 pgs, \$29.95
Volume 2, 400 pgs, \$29.95
- "Apple //gs ProDOS 16 Reference"
Disk included, 200 pgs, \$39.95

I expect Gary Little's new book, "Inside the Apple //gs", to be coming out by next March or April. No doubt there will be many more books coming out. Apple has a way of triggering whole new industries....

Many times we want to call "time out" during an assembly, for various reasons. Maybe a program has outgrown the available disk space and we need to swap source disks, or maybe we need to check the value of a label during assembly. We can't manually pause an assembly at a specific place during pass one, and it's difficult to do during pass two if the listing is on, since you have to sit and stare at the screen to tell where the assembly is. What we need is a PAUSE directive to tell the assembler to stop and wait for a keypress before continuing.

Back in May of 1983 Mike Laumer wrote up such a directive for the S-C Macro Assembler. The Assembler provides a .US directive for just such cases and Mike supplied the routines to use a line like .US SWAP SOURCE DISK to pause the assembly and display "SWAP SOURCE DISK" on the screen in inverse text. The Macro Assembler, Apple computers, and people's expectations have all changed in the last 3 1/2 years, so it seems like time to update and expand that article.

The .US vector normally contains JMP CMNT, a jump to the assembler's comment routine to just list a line. We can patch in the address of our handler and then have our code exit to CMNT when we're through. When control transfers to the .US vector the source line is in the system input buffer at \$200 and location \$7B contains an index into the buffer, pointing at the first character following the ".US" (normally a space). Another assembler variable that can come in handy for a .US feature is PASS, at \$60. This location contains a 0 during pass one of assembly and a 1 during pass two.

It only takes a few changes to adapt Mike's code to the Version 2.0 Macro Assemblers. The .US vector is now at \$D015 (\$8015 for ProDOS), so we have to change that .Equate line. CMNT has shifted around between various releases of Version 2.0, so I redid the code in INSTALL to transfer the correct address for CMNT out of the vector before installing PAUSE.

We only need to make two more changes to create a ProDOS version: alter the .US vector definition as shown in lines 1230-1240; and delete lines 1180-1190, 1290-1300, and 1400, since we don't need to worry about enabling/disabling the "Language Card" memory under the ProDOS assembler.

Most of the changes have to do with accomodating the //e 80-column display, with its division between main and auxiliary memory. I kept the technique of using the Y-register to index through the source line, and the X-register to index through screen memory. Toggling the Carry bit keeps track of which bank we need to store into, and incrementing Y after each store and incrementing and testing X after every other store takes care of the different indexes we need.

I thought we were just about done when I realized that this program wouldn't properly handle lower case text in the message string. To use inverse lower case we have to take the AltChar soft switch into account and adjust the ASCII values. I added

To boldly go at speeds no man has gone before.

*With the TransWarp™ accelerator, your IIe™ or II+™
is 40% faster than the IIgs!*

Computing at warp speed!

It's an experience you shouldn't miss. And with TransWarp, you won't have to. Because TransWarp will run your IIe or II+ software *3.6 times faster*—that's 40% faster than Apple's IIgs!

No more yawning while your Apple™ slowly rearranges text or calculates spreadsheets. With 256K of ultra fast RAM, TransWarp speeds up *all* Apple software—including AppleWorks, and all educational software, graphics and games. And it's compatible with all standard peripheral cards (such as RamWorks and Apple memory cards), hard disks, 3 1/2" disks, 80-column cards, modems, clock cards, mice and more! You name it, TransWarp accelerates it. There's even a 16 bit upgrade chip available.



"TransWarp is great! I have replaced all my other accelerators with it!"

Steve Wozniak, the creator
of Apple Computer

An important difference.

With TransWarp, software runs up to three times faster than with other cards, since the others can't accelerate programs in auxiliary memory. That's why TransWarp is so much faster than the rest. Nearly all of today's more powerful programs run partially or completely in auxiliary memory: programs like AppleWorks, Pinpoint, Managing Your Money, SuperCalc 3a, BPI and Pascal, to name a few. Why settle for a card that accelerates only part of the memory?

There's one more important difference. Since TransWarp doesn't use memory caching, you get *consistent* high speed performance.

A cinch to use.

Simply plug TransWarp into any slot in your Apple II, II+ or IIe—including slot 3 in the IIe. Instantly you'll be computing

at speeds you only dreamed about before. And should you ever wish to run at normal speed, simply press the ESC key while turning your Apple on.

Since TransWarp is completely transparent, you won't need preboot disks or special software. It's ready to go right out of the package!

Speed = Productivity

Imagine the productivity gains you'll achieve when your programs are running over three times faster. TransWarp is so powerful, your Apple will make IBM PCs™ and even ATs™ look like slowpokes.

- 3.6 MHz 65C02
- 256K of ultra fast on-board RAM
- Accelerates main *and* auxiliary memory
- Low power consumption for cool operation
- Totally transparent operation with all software
- Plugs into any slot, including slot 3 on the Apple IIe
- Accelerated 16 bit option available
- 5-year warranty

Satisfaction guaranteed!

Give *your* Apple the TransWarp advantage. With our risk-free 15-day money back guarantee, you have nothing to lose but wasted time. Call today!

TransWarp accelerator
16 bit upgrade (may add later)

\$279
\$89

For fast response:

Call Applied Engineering, 9 a.m. to 11 p.m., 7 days at (214) 241-6060. MasterCard, VISA and C.O.D. welcome. Texas residents add 5% sales tax. Add \$10.00 if outside U.S.A. Or mail check or money order to Applied Engineering, P.O. Box 798, Carrollton, TX 75006.

AE Applied Engineering
The Apple enhancement experts.

P.O. Box 798, Carrollton, TX 75006 (214) 241-6060

that code in at the last minute, and left it with odd line numbers and lower case opcodes, so you can see exactly how much extra effort it takes to deal with inverse lower case. If you're always going to use upper case text in your Pause messages you can save 24 bytes by leaving out those lines.

This program is specifically for the Apple //e 80-column display. For 40-column display you can just change the addresses in Mike's original article. For other 80-column displays you will probably have to give up some transparency, since you are unlikely to be able to display something on the screen without going through the usual I/O hooks. Maybe you can store directly into the card's memory, if the manufacturer documents how to do it.

```

1000 *SAVE S.NEW.PAUSE
1010 *-----
1020 * .US DIRECTIVE TO PAUSE DURING ASSEMBLY
1030 *
1040 *      SYNTAX: .US <phrase>
1050 *      RESULT: Displays <phrase> in inverse text
1060 *               and waits for a keypress
1070 *
1080 *-----
7B-      1090 CHAR.PTR .EQ $7B
1100
0200-    1110 WBUF      .EQ $200
07D0-    1120 CORNER    .EQ $7D0
1130
C000-    1140 KEYBOARD  .EQ $C000
C00E-    1141 alt.off   .eq $c00e
C00F-    1142 alt.on    .eq $c00f
C010-    1150 STROBE    .EQ $C010
C01E-    1151 alt.read  .eq $c01e
C054-    1160 PAGE1     .EQ $C054
C055-    1170 PAGE2     .EQ $C055
C080-    1180 PROTECT   .EQ $C080
C083-    1190 ENABLE    .EQ $C083
1200
FBE2-    1210 BELL      .EQ $FBE2
1220 *-----
D015-    1230 USR.VECT   .EQ $D015      DOS 3.3
1240 *               $8015      ProDOS
1250 *-----
1260      .OR $300
1270
1280 INSTALL
0300- AD 83 C0 1290      LDA ENABLE      write enable
0303- AD 83 C0 1300      LDA ENABLE      RAM card
0306- A2 01 1310      LDX #1            start with hi-bytes
0308- BD 16 D0 1320      .1 LDA USR.VECT+1,X get SC.CMNT address
030B- 48 1330      PHA                    stash it
030C- BD 65 03 1340      LDA EXIT+1,X    get PAUSE address
030F- 9D 16 D0 1350      STA USR.VECT+1.X set .US vector
0312- 68 1360      PLA                    recover stash
0313- 9D 65 03 1370      STA EXIT+1.X    set exit address
0316- CA 1380      DEX                    now do lo-bytes
0317- 10 EF 1390      BPL .1
0319- AD 80 C0 1400      LDA PROTECT     protect card
031C- 60 1410      RTS
1420 *-----
031D- A2 00 1430 PAUSE LDX #0            start at beginning of screen line
031F- 18 1440      CLC                    clear toggle
0320- AD 1E C0 1441      lda alt.read    get altchar status
0323- 08 1442      php                    stash it
0324- 8D 0F C0 1443      sta alt.on    altchars on
0327- A4 7B 1450      LDY CHAR.PTR      index into source line

```

0329- B9 00 02	1460 .1	LDA WBUF,Y	get char from call line
032C- F0 22	1470	BEQ .3	.EQ. is end of line
032E- 08	1471	php	preserve carry
032F- C9 60	1472	cmp #' '	test for lower case
0331- 29 3F	1480	AND #%00111111	invert char
0333- 90 02	1481	bcc .15	.CC. if upper case
0335- 09 40	1482	ora #%01000000	correct inverse lower case
0337- 28	1483 .15	plp	restore carry
0338- B0 0A	1490	BCS .2	branch if odd screen position
	1500		
033A- 8D 55 C0	1510	STA PAGE2	even, so use aux memory
033D- 9D D0 07	1520	STA CORNER,X	show character
0340- C8	1530	INY	next message character
0341- 38	1540	SEC	set toggle
0342- B0 E5	1550	BCS .1	always
	1560		
0344- 8D 54 C0	1570 .2	STA PAGE1	odd, so use main memory
0347- 9D D0 07	1580	STA CORNER,X	show character
034A- C8	1590	INY	next message character
034B- E8	1600	INX	next screen position
034C- E0 28	1610	CPX #40	line full?
034E- 90 D9	1620	BCC .1	no, get another char, clear toggle
	1630		
0350- 20 E2 FB	1640 .3	JSR BELL	beep
0353- AD 00 C0	1650 .4	LDA KEYBOARD	
0356- 10 FB	1660	BPL .4	wait for keypress
0358- 8D 10 C0	1670	STA STROBE	
035B- 8D 0E C0	1671	sta alt.off	assume altchars off
035E- 28	1672	plp	get altchar status
035F- 10 03	1673	bpl exit	.PL. if altchar was off
0361- 8D 0F C0	1674	sta alt.on	set altchars on
0364- 4C 1D 03	1680	EXIT JMP PAUSE	address modified by INSTALL
	1690		

RAMWORKS™

**ACCEPT NO SUBSTITUTES.
BECAUSE THERE AREN'T ANY.**

There's only one card like RamWorks. We've got the best hardware design. We supply the best software and we've got the best support from software companies.

If someone tempts you with an imitation, please get both sides of the story. You'll discover why RamWorks offers the best enhancements to AppleWorks and other programs, and at the lowest price.

GUARANTEED!

**214-241-6060
9 AM - 11 PM**

AE
"We Set the Standard"
APPLIED ENGINEERING

There are 256 bytes of RAM inside the clock chip in the Apple //gs. These bytes are backed up by the same battery that keeps the clock ticking when you turn off your Apple. You can read and write the battery RAM locations, but not the same as regular RAM. You can either do it the hard way, by direct hardware, or you can do it through the built-in firmware.

First, the easy way. When you turn on your //gs, the power-up routines install a lot of stuff in RAM in banks \$E0 and \$E1. At the beginning of \$E1 there are a lot of JMP opcodes, with long (24-bit) addresses. The one at \$E10000 is a jump to the Tool Locator. The Tool Locator is simply a way to access a lot of firmware subroutines without knowing their actual addresses. Instead of calling a firmware subroutine directly, you load up a subroutine number in a register and call the single known address, \$E10000.

To keep things organized, the //gs firmware designers require you to call \$E10000 with the 65816 in Native mode, with a JSL \$E10000. Any parameters the subroutines need must be pushed onto the stack before the JSL, and any results will be on the stack when the subroutine is finished. The carry status will indicate whether the subroutine returned an error code or not, just as in ProDOS MLI. If carry is clear, there was no error; if carry is set, there was an error and the error code is in the A-register. Regardless of the setting of the m- and x-status bits when you call \$E10000, it will return with both of them zero (full 16-bit mode).

You tell the Tool Locator which "tool" to call by a code number in the X-register. This is a 16-bit value, so you must have 16-bit mode on for the X-register when you call \$E10000 (x-status bit=0). It doesn't matter whether m-status is 0 or 1. The tool code is made up of a tool set number (00-FF, in the low byte) and a tool number (00-FF, in the high byte). The tool code to read all 256 bytes of battery RAM is \$0A03; to write 256 bytes out to battery RAM, the tool code is \$0903. The following program will read battery RAM:

```

                                1020      .OP 65816
                                1030      *-----
000800- 18                    1040 R      CLC
000801- FB                    1050      XCE
000802- C2 30                 1060      REP #$30
                                1070      *-----
000804- F4 00 00              1080      PEA BUF/256/256
000807- F4 00 09              1090      PEA BUF
00080A- A2 03 0A              1100      LDX ###0A03  READ BATTERY RAM
00080D- 22 00 00 E1          1110      JSL $E10000
                                1120      *-----
000811- 38                    1130      SEC
000812- FB                    1140      XCE
000813- 60                    1150      RTS
                                1160      *-----
000814- 18                    1170 W      CLC
000815- FB                    1180      XCE
000816- C2 30                 1190      REP #$30
                                1200      *-----
000818- F4 00 00              1210      PEA BUF/256/256
00081B- F4 00 09              1220      PEA BUF
00081E- A2 03 09              1230      LDX ###0903  WRITE BATTERY RAM
000821- 22 00 00 E1          1240      JSL $E10000
                                1250      *-----

```


Save Time Debugging Your Assembly Language Programs With QUICK TRACY

Now you can whiz out your assembly language programs even faster than before with **QUICK TRACY**. This new debugger was "just the thing Bill needed while working on the UniDisk boot program", in the July 1986 issue of Apple Assembly Line. **QUICK TRACY** will rapidly trace through your machine language object code, suggesting the execution of JSR instructions in real-time. Even though **QUICK TRACY** suggests, you are in complete control at all times. **QUICK TRACY** extends the S-C Macro Assembler by providing you with 27 powerful debugging commands. Whether you have the DOS 3.3 version or the ProDOS version, **QUICK TRACY** can do the job for you. Whether you have an Apple //e or Apple][+ in 40 or 80 columns, **QUICK TRACY** can do the job for you. **QUICK TRACY** can execute those parts of your program at real-time that you know are okay, while carefully tracing and displaying all registers and flags for each instruction in those areas of your program that you designate. **QUICK TRACY** has 13 independent conditional breakpoints that you may set to SPELL OUT TROUBLE. One major advantage is that **QUICK TRACY** is located in memory with the S-C Macro Assembler; so as soon as you finish assembling your program, it is ready for **QUICK TRACY** to help you find the bugs. Let's face it. Even the very best programmers get bugs, and they have debuggers for their hardware. Now here is one specifically designed for your hardware and software needs.

Why are you wasting your valuable time debugging when you could have **QUICK TRACY** doing that tedious work for you?

QUICK TRACY is available in either DOS or ProDOS for only \$35.00
(or both versions for only \$55.00).

T'n'T SOFTWARE
624 Charmain Drive
Campbell, CA 95008
(408) 378-8358

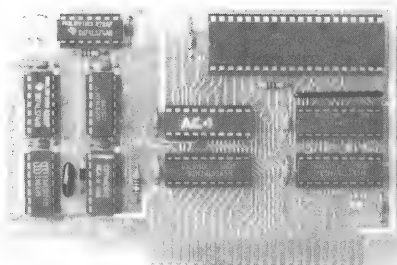
We Accept Visa or MasterCard.

For Postage and Handling, please include \$2.00 if U.S., \$5.00 for all other countries.
California state residents, please include applicable sales tax.

QUICK TRACY requires S-C Macro Assembler version 2.0, DOS 3.3, or ProDOS,
Apple][+ or Apple //e, at least 64K of RAM, printer and disk drive.

"Apple" is a registered trademark of Apple Computer, Inc.

With the Z-80 Plus®, run over 5000 new CP/M® programs.



*Now, get two computers
in one, and all the
advantages of both.*

Enter the CP/M world with the new Z-80 Plus card from Applied Engineering, and introduce your Apple IIe, IIgs or II+ to thousands of CP/M programs. Only the Z-80 Plus comes standard with the new 5.1 software, the most advanced system ever for running CP/M programs.

The new 5.1 boasts advanced features like built-in disk emulation for popular memory expansion boards, boosting both system speed and storage capacity. And menu-driven utilities that let you get to work faster. The Z-80 Plus also lets you run older CP/M programs—all the way down to Version 1.6 (2.2 is the most popular).

The Z-80 Plus is the only card on the market capable of accessing more than 64K. If you have an extended 80-column card, all 128K is usable. And if you have RamWorks, RamFactor, αRam, αRam Plus, or an Apple memory card in your IIe, IIgs or II+, up to 1088K is available.

Each Z-80 Plus comes with our CP/M Ram Drive software, enabling most memory cards to work as a high-speed Ram disk so CP/M software can run up to *twenty times faster*. So packages like WordStar, dBASE II, Turbo Pascal and most others run at blinding speed.

Simply plug the Z-80 Plus into any slot in your Apple. You'll get the benefits of two computers in one—all at an unbelievably low price (only \$149!).

- Fully compatible with ALL CP/M software
- Fully compatible with 5¼" *and* 3½" disk drives as well as most hard disks including the Sider
- Fully compatible with Microsoft disks (no pre-boot required)
- Specifically designed for high speed operation
- Runs WordStar, dBASE II, Turbo Pascal, and ALL other CP/M software with *no pre-boot*
- Semi-custom I.C. and low parts count allows Z-80 Plus to fly through CP/M programs with extremely low power consumption (we use the Z-80B)
- Does EVERYTHING other Z-80 boards do, *plus* Z-80 interrupts
- Five-year warranty

To get your Z-80 Plus or for more information see your dealer, or call 214-241-6060 9 a.m. to 11 p.m. seven days, or send check or money order to Applied Engineering, MasterCard, VISA and C.O.D. welcome. Texas residents add 5½% sales tax. Add \$10.00 outside U.S.A.

AE Applied Engineering™
P.O. Box 798, Carrollton, TX 75006
(214) 241-6060

CP/M is a registered trademark of Digital Research

```

000825- 38          1260          SEC
000826- FB          1270          XCE
000827- 60          1280          RTS
                        1290  *-----
0900-          1300 BUF      .EQ $900
                        1310  *-----

```

When I did this on my //gs prototype, this is what I got:

```

0900-00 00 00 01 00 00 0D 06 02 01 01 00 01 00 00 00-
0910-00 00 07 06 02 01 01 00 00 00 0F 07 00 08 0B-
0920-01 01 00 00 00 00 01 01 05 00 00 00 03 02 02-
0930-00 00 00 00 00 00 0C 08 00 01 02 03 04 05 06-
0940-07 0A 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D-
0950-0E 0F FF FF FF FF FF FF FF FF FF FF FF FF FF-
0960-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
0970-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
0980-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
0990-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
09A0-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
09B0-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
09C0-C2 CF C2 A0 D3 C1 CE C4 C5 D2 AD C3 C5 C4 C5 D2-BOB SANDER-CEDER
09D0-CC CF C6 A0 FF FF FF FF FF FF FF FF FF FF FF-LOF
09E0-FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF-
09F0-FF FF FF FF FF FF FF FF FF FF FF FF FF FF 27 CE 8D 64-'N.d

```

Those last four bytes are some kind of a check sum, handled automatically by the tool. I suppose that if the checksum is incorrect on power up, you will be popped into the configurator instead of going into a boot. You can read these bytes, but you cannot write them with the tool: the tool will calculate a checksum and write it when you write the other 252 bytes. Bytes \$52 through \$FB are either used by the operating system or reserved for the future. Just for fun, I have now written my own name in ASCII code into the bytes starting at \$C0.

The rest of the bytes are used as shown in the following table. Where two choices are shown, separated by a slash, the left one has a code of \$00 and the right choice has a code of \$01.

Port 1	Port 2	
\$00	\$0C	Printer/Modem
\$01	\$0D	Line Length (Any/40/72/80/132)
\$02	\$0E	Delete LF after CR (No/Yes)
\$03	\$0F	Add LF after CR (No/Yes)
\$04	\$10	Echo (No/Yes)
\$05	\$11	Buffer (No/Yes)
\$06	\$12	Baud Rate
\$07	\$13	Data & Stop Bits
\$08	\$14	Parity
\$09	\$15	DCD Handshake (No/Yes)
\$0A	\$16	DSR Handshake (No/Yes)
\$0B	\$17	XON-XOFF Handshake (No/Yes)

Display Parameters

\$18	Color/Monochrome
\$19	40/80 Column
\$1A	Text Color (00-0F)
\$1B	Background Color (00-0F)
\$1C	Border Color (00-0F)
\$1D	60/50 Hertz Operation
\$29	Text Language (0=English)
\$2F	Flash Rate

Keyboard Parameters

\$2A Language (0=English)
\$2B Buffering (No/Yes)
\$2C Repeat Speed
\$2D Repeat Delay
\$30 Shift Caps-LowerCase (No/Yes)
\$31 Fast Space-Delete Keys (No/Yes)
\$32 Dual Speed (Normal/Fast)

Slot Configuration

\$21-27 Slot 1-7 Internal/External
\$28 Boot Slot

Miscellaneous

\$1E User Volume
\$1F Bell Volume
\$20 System Speed (Normal/Fast)
\$2E Double-Click Time
\$33 High Mouse Resolution
\$34 Date Format
\$35 Time Format
\$36 Min RAM for Ramdisk
\$37 Max RAM for Ramdisk
\$38-40 Count & Languages
\$41-51 Count & Layouts
\$80 AppleTalk Node Number
\$81-A1 Operating System Variables

It is possible, as I said before, to talk directly to the battery RAM via I/O addresses. If you learn how to do this, and you use the skill to write values into battery RAM, you will probably do so without properly changing the checksum. In that case you have violated your system, and your next power-up will revert to default values for all parameters. It will stay that way until you reconfigure everything and/or install a proper checksum. The best policy is to use the standard firmware tools for all reading and writing, so that the checksum stays current.

You do not have to read or write the whole battery RAM at once. There are two tools for reading and writing a single byte. Tool Code \$0B03 will write one byte, and tool code \$0C03 will read one byte. The following code segments illustrate how to do it. The code as shown must be in Native Mode, with both x- and m-bits zero (full 16-bit mode).

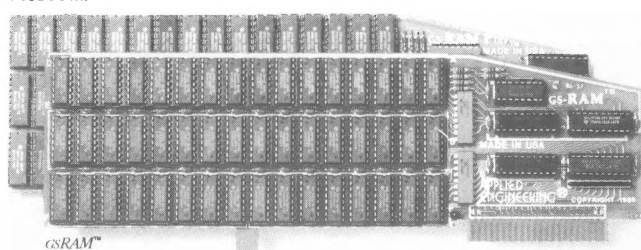
```
WR  PEA $00xx      xx is new value for byte
    PEA $00yy      yy is address in battery RAM
    LDX ##$0B03    write xx at yy
    JSL $E10000

-----
RD  PEA $0000      make room for result
    PEA $00yy      yy is address in battery RAM
    LDX ##$0C03    read value at yy
    JSL $E10000
    PLA            get result from stack (00xx)
```

Insist on GSRAM™ When You Buy Your IIGs™

*Expand the IIGs RAM and ROM with the GSRAM or GSRAM Plus
with ROM Pak. Available now with 256K to 8 MEG!*

GSRAM Plus™



Remember the 16K cards for the II+ and the 64K cards for the IIx? At the time, that much memory seemed like a lot. But when the owners of these memory cards came to us for more memory, many had to throw away their smaller Apple memory cards or try to sell them. Most of our customers told us that had they known about Applied Engineering's larger memory cards when they bought their Apple, they would have purchased them at the same time.

GSRAM and GSRAM Plus are available now, allowing up to 8 MEG of memory expansion. That's 8 times the memory capacity of Apple's card and just look at the benefits that only GSRAM and GSRAM Plus have over Apple's card:

- Lower cost
- Has 6 RAM banks (Apple's card has 4)
- Has memory expansion port
- Has ROM expansion port
- No configuration blocks to set
- No soldered in RAM chips
- Expandable to 8 MEG
- Expands AppleWorks internal limits
- Built-in Hi-Res self-diagnostic software
- 5 year hassle free warranty (Apple has a 90 day warranty)
- Made in USA

GSRAM for More AppleWorks Power

Only GSRAM and GSRAM Plus eliminates AppleWorks internal memory limits, increasing the maximum number of records available from 6,000 to over 25,000 and only GSRAM and GSRAM Plus increases the number of lines permitted in the word processing mode from 6,000 to over 15,000. And only GSRAM and GSRAM Plus offers a built-in printer buffer so you can continue using AppleWorks while your printer is printing. GSRAM and GSRAM Plus even expand the number of lines in the clipboard from 255 to 2047 and will auto segment large files so they can be saved on two or more disks. You can

even have Pinpoint or Macroworks and your favorite spelling checker in RAM for instant response. GSRAM and GSRAM Plus will even display the time and date right on the AppleWorks screen. Nothing comes close to enhancing AppleWorks so much.

Turn Your IIGs into a Giant

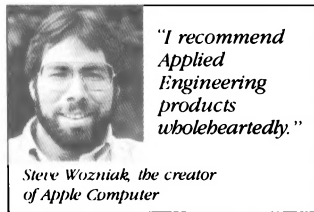
Simply plug GSRAM into the IIGs memory expansion slot and you've got up to 8 megabytes of RAM at your fingertips—all of it instantly and automatically recognized by the IIGs. GSRAM is compatible with all IIGs software, including AppleWorks, as well as BASIC®, ProDOS, DOS 3.3, PASCAL®, "C" and CP/M®.

Grow by Bytes or Megabytes

We offer GSRAM in two configurations so you can increase your memory 256K at a time (GSRAM) or a megabyte at a time (GSRAM Plus). Both offer full compatibility, lower cost than other boards, and easy expandability. And both are extremely low in power consumption. A fully expanded GSRAM operates at only 375 ma, and GSRAM Plus at only 270 ma (even with 6 megabytes on board!)

GSRAM—for Normal Memory Requirements

GSRAM is available with 256K, 512K, 1 MEG or 1.5 MEG of memory already on board. If you don't need the full 1.5 MEG now, you can choose a GSRAM with less memory and expand it up to 1.5 MEG in the future—or upgrade to GSRAM Plus for a small change.



*"I recommend
Applied
Engineering
products
wholeheartedly."*

*Steve Wozniak, the creator
of Apple Computer*

With an optional piggyback card, you can expand GSRAM even higher than 1.5 MEG! (Other cards are only expandable to 1 MEG.)

GSRAM Plus—for Growing by Leaps and Bounds

GSRAM Plus is the first Apple memory card to use 1 MEG RAM chips on the main board. It's available with 1 to 6 MEG on board. If you don't need the whole 6 MEG now, you can buy a GSRAM Plus with less memory and easily expand it in the future.

GSRAM Plus can be expanded up to 8 MEG with an optional piggyback card.

Easy Expansion

Both GSRAM and GSRAM Plus use standard RAM chips that are readily available and just plug right in. So unlike other cards, you'll find expanding your GSRAM or GSRAM Plus easy, convenient and very economical. And with our optional ROM expansion module you can even increase the IIGs's ROM space and all in just one slot.

Quality and Support of the Industry Leader

Applied Engineering is the largest supplier of Apple peripherals in the world. We invented the first large RAM cards for the Apple. With a 5-year "no-hassle" warranty and outstanding technical support, you can be sure GSRAM and GSRAM Plus will deliver the performance you're looking for—or return them within 15 days for a full refund.

GSRAM with 256K	\$169
GSRAM with 512K	\$219
GSRAM with 1 MEG	\$299
GSRAM with 1.5 MEG	\$379
GSRAM with 2-8 MEG	CALL
GSRAM Plus with 1-8 MEG	CALL

Order today!

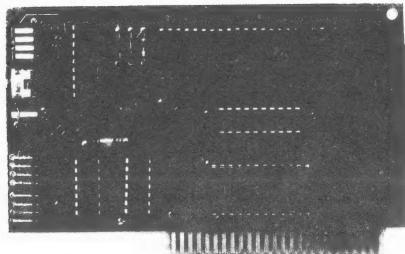
See your dealer or call Applied Engineering today, 9 a.m. to 11 p.m. 7 days. Or send check or money order to Applied Engineering, MasterCard, VISA and C.O.D. welcome. Texas residents add 5½% sales tax. Add \$10.00 outside U.S.A.

AE APPLIED ENGINEERING™
The Apple enhancement experts.
(214) 241-6060

P.O. Box 798, Carrollton, TX 75006

CP/M is a registered trademark of Digital Research, Inc.

The first clock card to add time and date to AppleWorks™ screens and files.



Timemaster H.O. So many features its become the world's best selling clock card.

It's the smart way to put the time and date on your Apple IIe* or II+*. Because only the Timemaster H.O. packs *ALL* the important features of all the competition *combined*, including leap year, year (not just in PRO-DOS), month, date, day of week, hours, minutes, seconds and milli-seconds. Its totally PRO-DOS, DOS 3.3, PASCAL and CP/M compatible. And, of course, it works better than any other clock with AppleWorks.

With the Timemaster H.O., you're assured of reliable, trouble-free operation along with simple, one command time setting, accuracy through leap years, and a firmware maintained interrupt setting.

If you're using or writing software for other clock cards, you're still covered. Because the H.O. will *automatically* emulate them. And only the Timemaster H.O. adds 15 new commands to BASIC. The H.O. even comes complete with two disks full of sample programs, including a computerized appointment book, a DOS dating program, interrupt programs, and over 30 programs that others charge you extra for...or don't even offer. As a low cost option, you can add true BSR remote control to the H.O. giving you remote control of up to 16 lights and appliances in your home or office.

- Time in hours, minutes, seconds and milliseconds, date with year, month, day of week and leap year • 24 hour military format or 12-hour AM/PM format • Eight software controlled interrupts so you can run two programs at the same time (many examples included) • Can be set to the second • Allows AppleWorks to time and date stamp all data automatically • Displays time and date on AppleWorks screens and files • Five year warranty

Timemaster H.O. \$129.00
BSR option (may be added later) \$ 59.00

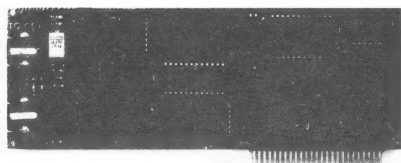
Call to order today, 9 a.m. to 11 p.m. seven days, or send check or money order to Applied Engineering, Master Card, VISA and C.O.D. welcome. Texas residents add 5% sales tax. Add \$10.00 outside U.S.A.

AE Applied Engineering
The Apple enhancement experts.

P.O. Box 798, Carrollton, TX 75006 (214) 241-6060

Viewmaster 80

Now run AppleWorks™ on your II+ and keep full 80 column compatibility!



	Apple IIe compatible	Apple II+ compatible	Apple IIc compatible	Apple IIcx compatible	Apple IIx compatible	Apple IIg compatible	Apple IIgs compatible	Apple IIx compatible	Apple IIc compatible	Apple IIgs compatible
VIEWMASTER 80	x	x	x	x	x	x	x	x	x	x
SI PRTERM	x	x	x	x	x	x	x	x	x	x
WIZARD 80	x	x	x	x	x	x	x	x	x	x
VISION 80	x	x	x	x	x	x	x	x	x	x
OMNIVISION	x	x	x	x	x	x	x	x	x	x
VIEWMAX 80	x	x	x	x	x	x	x	x	x	x
SMARTERM	x	x	x	x	x	x	x	x	x	x
VIDEOTERM	x	x	x	x	x	x	x	x	x	x

One look at the chart will give you some of the reasons there's only one smart choice in 80 column cards for your Apple. But the real secret to Viewmaster 80's success is something even better: Total compatibility.

Each Viewmaster 80 includes our powerful AppleWorks expand software, allowing AppleWorks to run on the II+ with only 64K (or more) memory. (We recommend the RamFactor memory card, but any compatible Apple memory card will work.) The software provides our full range of AppleWorks enhancements, including expanded records, word processor, multi-disk saving, time and date display on screen with any PRO-DOS clock, and more!

The Viewmaster 80 works with all 80 column applications, including DOS 3.3, PRO-DOS, CP/M, Pascal, WordStar, Format II, Applewriter II, dBase II, Visicalc, Multiplan, and *hundreds* of others—including AppleWorks.

Here are just a few of the powerful features the Viewmaster 80 delivers for a great price (\$139).
 • 80 Characters by 24 lines • Fully compatible with all Apple languages and software • Highest compatibility with existing 80 column software • Very low power consumption • High speed (18 MHz) scroll rate • Upper and lower case characters and true descenders, both inverse and normal; all on-screen editing functions are supported • User-definable cursor shape • Compatible with Apple II, II+ and IIe • Five-year warranty

Call today to order or for more information, 9 a.m. to 11 p.m. seven days, or send check or money order to Applied Engineering, MasterCard, VISA and C.O.D. welcome. Texas residents add 5% sales tax. Add \$10.00 outside U.S.A.

AE Applied Engineering
P.O. Box 798, Carrollton, TX 75006
(214) 241-6060

The Clock/Calendar Chip not only contains the battery RAM; it also contains the date and time information, naturally. There are three tools for reading and writing the time and date. You can read time/date in either hexadecimal format or as an ASCII string, and you can write a new time/date in a hex format. The following code segments illustrate how to use the tools.

Read.Time.Into.ASCII.String

```
PEA BUFFER/256/256  Hi 16-bits of buffer address
PEA BUFFER          Lo 16-bits of buffer address
LDX ##$0F03        Tool Code
JSL $E10000
```

The date and time will be converted to ASCII (with msb = 1) and stored in BUFFER, according to the formats selected in the configuration menu (stored in battery RAM locations \$34 and \$35). The most likely choice among North Americans will be the format "mm/dd/yy HH:MM:SS xM", but you have five other possibilities.

Read.Time.Hex

```
PEA 0      Make room for 8 bytes
PEA 0      to be returned
PEA 0
PEA 0
LDX ##$0D03
JSL $E10000
PLA      Get $MMSS (minutes, seconds)
STA MMSS
PLA      Get $yyHH (year, hours)
STA YYHH
PLA      Get $mddd (month, day)
STA MMDD
PLA      Get day of week (in low byte)
STA DOW
```

The value for day of week runs from 0 to 6, with 0=Sunday. The value for "day" is 0-30, meaning that you have to add 1 to get the true day number. (Why? This is a little ridiculous!) Likewise, the value for month is 0-11, with 0 standing for January. (I can understand why the hardware might work with 0-based values for day and month, but why couldn't the firmware do the correction to "real" day and month numbers?) The year is specified as the actual year number minus 1900. I hope that means my //gs will still give correct dates after 1999. If the value of the "yy" byte can go all the way to 255, then we could use //gs until the end of the year 2155. Frankly, I think I'll get tired of computers before then.

To write a new date and time out to the Clock chip, you have to push the values onto the stack and call the tool:

Write.New.Time

```
PEA $mddd      month, day
PEA $yyHH      year, hour
PEA $MMSS      minute, second
LDX ##$0E03
JSR $E10000
```

Again, the month and day values are zero-based. Note that you cannot update the day-of-week directly; apparently it is only a CALCULATED value provided when you READ the date/time in hex format.

You might wonder whether anyone would really NEED all the above information. After all, Apple has provided the configuration system to see/modify all those parameters. The problem is you cannot really use that system unless you can SEE. A lot of Apple owners are not able to see, so they use the ECHO or other some other brand of speech synthesizer to speak everything that goes out to the screen. The configuration program cannot be made to speak, as it is now written. Larry Skutchan is planning to write some sort of speaking version of the configurator, and the information above is just what he needs.

EnterSoft

Basic-like macros for the S-C Macro Assembler.

Available for the Apple II, II+, IIe, and IIfx.

Uses whatever DOS your S-C Macro Assembler uses.

Soon to be released for PRODOS.

Source code included with every disk!!!

Cost? A paltry \$30.00!!!

To Order - Send Check or Money Order to:



Mark Manning
c/o Baggy Games
P. O. Box 591894
Houston, Texas 77259-1894

Apple is a registered Trademark of Apple Computer Inc.

The S-C Macro Assembler is produced by The S-C Software Corporation.

EnterSoft is produced by Mark Manning/Simulacron I/Baggy Games

Apple Assembly Line (ISSN 0889-4302) is published monthly by S-C SOFTWARE CORPORATION, P.O. Box 280300, Dallas, Texas 75228. Phone (214) 324-2050. Subscription rate is \$18 per year in the USA, sent Bulk Mail; add \$3 for First Class postage in USA, Canada, and Mexico; add \$14 postage for other countries. Back issues are available for \$1.80 each (other countries add \$1 per back issue for postage). A subscription to the newsletter and the Monthly Disk containing all source code is \$64 per year in the US, Canada and Mexico, and \$87 to other countries.

All material herein is copyrighted by S-C SOFTWARE CORPORATION, all rights reserved. (Apple is a registered trademark of Apple Computer, Inc.)